uc3m | Universidad Carlos III de Madrid

Grado en Ingeniería Electrónica Industrial y Automática
Curso 2017-2018

Trabajo Fin de Grado

Diseño de una maqueta activada con EMG

Estefanía Alvar Sánchez

Tutor

María Dolores Blanco Rojas

Escuela Politécnica Superior de Leganés, 2018

Resumen

En este trabajo se presenta el diseño de una maqueta activada con señales de electromiografía, con el objetivo de utilizarla en una nueva práctica para los alumnos de la asignatura “Aplicaciones de la automática en biomédica”.

La adquisición de las señales se realiza con el Myo Armband, un dispositivo equipado con ocho electrodos capaz de reconocer un determinado número de movimientos de la mano y el brazo en que está colocado. Gracias a la posibilidad de modificar su software inicial en MATLAB, se procede a realizar la identificación de los gestos de apertura y cierre de la mano mediante la técnica de reconocimiento de patrones.

Por último, para que los alumnos puedan visualizar en un sistema físico real la clasificación realizada, se fabrica una pinza de tres dedos mediante impresión 3D, capaz de replicar los movimientos realizados por el usuario a tiempo real.

Palabras clave: electromiografía, redes neuronales, reconocimiento de patrones, tiempo real

Agradecimientos

En primer lugar, me gustaría dar las gracias a mi tutora, Dolores Blanco, a Álvaro Villoslada y a Dorin Copaci, por su ayuda durante el desarrollo de todo el proyecto.

Quiero dar las gracias especialmente a mi familia, por darme siempre todo su apoyo, no solo en mis estudios, sino en todo, y por hacerme confiar en que, con trabajo y esfuerzo, siempre puedo conseguir aquello que me proponga. Sin vosotros nunca hubiera llegado hasta aquí. Gracias.

Índice de contenidos

Índice de figuras

Índice de tablas

1. Introducción

1.1 Motivación

Este trabajo nace de la necesidad de realizar una nueva práctica para la asignatura de "Aplicaciones de la automática en biomédica", del cuarto curso del grado en Ingeniería Electrónica Industrial y Automática.

Esta asignatura tiene como objetivo que los alumnos descubran las diferentes aplicaciones que puede tener la automática en la medicina. En ella, se estudian los robots quirúrgicos, las prótesis, órtesis y exoesqueletos, además de diferentes técnicas de control para los mismos y los tipos de sensores y actuadores utilizados [1].

Por tanto, con este proyecto se pretende que, mediante una maqueta funcional sencilla, el alumno comprenda cómo es el proceso de adquisición de señales EMG y el uso de redes neuronales para la clasificación de movimientos.

Para llevarlo a cabo, se utiliza MATLAB, ya que es el software utilizado en el resto de prácticas de la asignatura y, por tanto, los alumnos ya están familiarizados con el mismo, permitiéndoles centrarse únicamente en el aprendizaje de los nuevos conceptos de la electromiografía y no en el aprendizaje de un nuevo lenguaje de programación.

1.2. Antecedentes

Este proyecto parte de un estudio previo realizado por Alejandro Sánchez Anillo en su Trabajo de Fin de Grado "Matriz de electrodos EMG para detección de intención de movimiento de la mano" [2]. En él, se comenzó por primera vez la investigación de las posibles aplicaciones del Myo Armband en el campo de la biomedicina. Este proyecto consistió en el análisis y comprensión del software inicial de MATLAB que trae el propio dispositivo. Además, se realizaron algunas ampliaciones de código que permitirían guardar los datos adquiridos y representarlos gráficamente, con el objetivo de visualizar las señales EMG generadas por diferentes movimientos de la mano y del antebrazo y, así, poder realizar un análisis comparativo de los mismos.

1.3. Objetivos

El presente trabajo tiene como objetivo principal el diseño de una maqueta funcional completa de bajo coste activada mediante señales EMG, capaz de detectar y clasificar movimientos sencillos tales como la apertura y cierre de la mano, replicándose a tiempo real en una pinza impresa. Además, tiene un fin didáctico, ya que se pretende que el diseño de esta maqueta sirva como elemento de apoyo al aprendizaje en la parte

práctica de la asignatura “Aplicaciones de la automática en biomédica”. Así, los alumnos podrán comprender el proceso de adquisición de datos EMG, su procesamiento y, de una manera visual, ver la clasificación de movimientos mediante una maqueta sencilla que replique los mismos.

Para llevarlo a cabo, se han pautado los siguientes objetivos específicos:

- Generalización del software inicial del Myo Armband que permita la conexión con MATLAB para poder trabajar con ello de manera sencilla
- Adquisición y procesamiento de señales EMG
- Identificación y clasificación de movimientos mediante la generación de una red neuronal
- Fabricación de pinza
- Integración del sistema de control con la pinza
- Funcionamiento a tiempo real

1.4. Entorno socio-económico

Los alumnos de los grados en Ingeniería requieren, además de una amplia base de fundamentos teóricos en todas las materias, un elevado nivel de la aplicación de esos conocimientos teóricos a la práctica para, así, poder desempeñar las funciones de un Ingeniero en su futuro en el mundo laboral.

En la Escuela Politécnica de la Universidad Carlos III de Madrid se valora notablemente esta formación práctica de los alumnos pero, debido a la limitación de recursos existente, es necesario que estas prácticas de laboratorio sean de bajo coste, primando siempre la importancia de las mismas, ya que es la manera de acercar a los alumnos a sistemas reales, más allá de la simulación.

En la ficha de la asignatura “Aplicaciones de la automática en biomédica”, se recogen los contenidos de la misma, entre ellos: prótesis y exoesqueletos robotizados, sistemas de control, así como robots quirúrgicos y equipos orientados a la rehabilitación [1].

El objetivo de las prácticas es que el alumno sea capaz de llevar a cabo el diseño de sistemas de control y, así, poder aplicarlo en el diseño de dispositivos automáticos.

Por ello, la maqueta propuesta en este proyecto ayudará en el aprendizaje de estos contenidos, en concreto, en la adquisición y análisis de señales y su aplicación en el control de prótesis.

1.5. Marco regulador

En este apartado, se recoge la normativa aplicable al proyecto:

- SENIAM ("Surface ElectroMyoGraphy for the Non-Invasive Assessment of Muscles") [3]:
  - En cuanto al tamaño de los electrodos, recomienda que todos ellos sean del mismo. En Europa se prefiere un tamaño de 1 cm, como es el caso del Myo Armband.
  - En lo referente a la distancia entre los electrodos, entendida como la distancia entre los centros de cada uno de ellos, recomienda una distancia de 2cm. En este caso, el Myo Armband presenta una distancia de alrededor de 2.5cm, cumpliendo por tanto con la distancia mínima recomendada.
  - En lo relativo a la localización, recomienda que estos estén en el sentido de las fibras musculares, tomando la muñeca como punto para el electrodo de referencia, tal y como es colocado el Myo Armband.

- Directiva 93/42/CEE [4]: esta normativa diferencia en cuatro grupos los dispositivos médicos. En este caso, el Myo Armband cumple con esta normativa dentro del grupo I, al ser un producto no invasivo

1.6. Estructura de la memoria

Este trabajo está organizado en diferentes capítulos, cuyo contenido se detalla a continuación.

- Capítulo 2 *Estado del Arte.* Se explican los diferentes métodos de activación de prótesis y exoesqueletos (2.1), así como diferentes dispositivos existentes que utilizan las señales EMG (2.2). En último lugar, se describen diferentes prácticas docentes relacionadas con la electromiografía (2.3).

- Capítulo 3 *Detección de movimiento en función de la señal EMG.* Este capítulo cuenta con tres secciones. La primera, en la que se explica la adquisición de la señal (3.1), en la que se da una introducción teórica acerca de las señales EMG así como la descripción del Myo Armband y, por último, se describe cómo se ha realizado la adquisición de datos para el presente trabajo. A continuación, se explica el procesamiento de las señales (3.2), comenzando por la segmentación, seguido de la extracción de características y terminando con la reducción de dimensionalidad. Y, la última sección, en la que se explica la clasificación de las señales (3.3), explicando los diferentes clasificadores que existen y la aplicación de redes neuronales en este proyecto.

- Capítulo 4 *Funcionamiento a tiempo real.* Se describe el código implementado para el uso de la pinza a tiempo real, incluyendo la programación del microcontrolador.

- Capítulo 5 *Fabricación de la pinza.* Se describen las diferentes partes de las que está compuesta la pinza y su fabricación.

- Capítulo 6 *Guion de prácticas.* En este capítulo se muestra un guion de ejemplo para el uso de la maqueta en prácticas docentes.

- Capítulo 7 *Resultados.*

- Capítulo 8 *Conclusiones.*

2. Estado del arte

2.1. Métodos de activación de prótesis y exoesqueletos e interfaces

A la hora de realizar una prótesis o un exoesqueleto, es importante elegir la interfaz con el paciente que se va a utilizar. En algunos casos, el control de alto nivel se realiza a partir de una interacción directa en la que se adquieren y analizan señales del cuerpo humano, como pueden ser las señales EEG (electroencefalograma) o las señales EMG (electromiografía), con el objetivo de determinar qué movimiento quiere realizar el usuario. En otros casos, la interacción puede realizarse a través de dispositivos con los que el usuario selecciona de forma explícita el movimiento a realizar.

En el primer grupo, se encuentra el control neural, basado en la adquisición de señales EEG. Las señales EEG provienen de la actividad eléctrica del cerebro. Estas señales son captadas mediante la colocación de electrodos en el cuero cabelludo. [5]

Por definición, la resolución espacial de las señales EEG es la capacidad de este método para representar gráficamente los lugares del cerebro activados mientras que, la resolución temporal, es la capacidad de detectar los cambios de la señal en el tiempo. A pesar de que los electroencefalogramas tienen una gran resolución temporal (de 1 a 3 milisegundos en función del hardware de adquisición) presentan una baja resolución espacial. Esta resolución espacial depende del número de electrodos que se sitúen, por lo que para mejorarla es necesario colocar un mayor número de los mismos. Esto implica que la técnica para adquirir las señales EEG no es práctica para el usuario y, por tanto, tampoco para utilizarla en el control de prótesis. [6]

Fig. 2.1: Neuroprótesis [7]

También, dentro de este primer grupo, se encuentra el control mioeléctrico, basado en el procesamiento de señales EMG, es decir, por la actividad eléctrica de los músculos. En el capítulo 3 se explican en profundidad las señales EMG puesto que son la base en que se fundamenta el presente trabajo.

En cuanto a las técnicas en las que se usan dispositivos para definir el movimiento deseado, se encuentra el control con tecnología inalámbrica. Son técnicas que permiten al usuario elegir el movimiento o la posición que desea realizar. Las prótesis con controlador MORPH ("Myoelectrically Operated RFID Prosthetic Hand") [8] se equipan con un lector de RFID. De esta manera, el paciente dispone de varias tarjetas RFID que le permiten elegir la posición de dicha prótesis en función del objeto que quiere coger. A pesar de englobarse dentro de técnicas que no requieren interacción directa con las señales del usuario, para detectar la intención de movimiento sí es necesario implementar un sensor EMG. La principal desventaja que presenta este tipo de prótesis es el número limitado de objetos que puede reconocer, así como la dificultad de utilizarla para objetos pequeños donde no se puede colocar una tarjeta RFID.

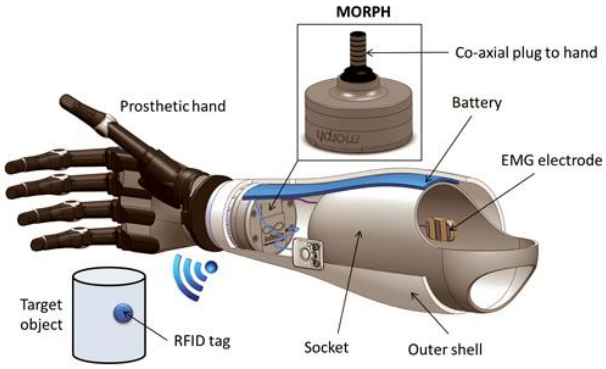


Fig. 2.2: Prótesis con controlador MORPH [9]

Otro método de control es el que utiliza tecnología Bluetooth. En general, el uso de esta tecnología se acompaña del desarrollo de aplicaciones móviles. Vinculando la prótesis, se pueden seleccionar las posiciones de la misma desde la pantalla del Smartphone [8].

Fig. 2.3: i-Limb [10]

Por último, se están desarrollando prótesis equipadas con cámaras. De manera que, mediante visión artificial, la propia prótesis detecta el objeto que tiene delante y se mueve en función de ello para adoptar la posición correcta. Además, se les ha dotado de inteligencia artificial, por lo que además de reconocer los objetos que tiene predeterminados, es capaz de aprender a reconocer nuevos objetos a medida que se va usando.

Fig. 2.4: Objeto reconocido por prótesis con visión y posición adoptada [11]

Como desventaja, en situaciones de poca iluminación donde los objetos no se pueden ver correctamente, podría no funcionar de manera adecuada.

Fig. 2.5: Prototipo de prótesis con visión artificial [12]

2.2. Dispositivos EMG

Como se ha mencionado en el apartado 2.1, una de las maneras de controlar prótesis es mediante señales EMG. A lo largo de la historia, se han desarrollado numerosos dispositivos de este tipo. La primera de ellas fue la de Reinhold Reiter [13] alrededor de 1940. Dado que para la fecha aún no se había inventado el transistor, esta prótesis utilizaba tubos de vacío. Por ello, no se podía transportar, ya que no poseía batería y, además, era sumamente pesada. Pero, a pesar de los inconvenientes, fue la primera prótesis en que el control de apertura y cierre de la mano se realizaba a través de la actividad eléctrica de un músculo y, por tanto, fue el origen de las prótesis mioeléctricas.

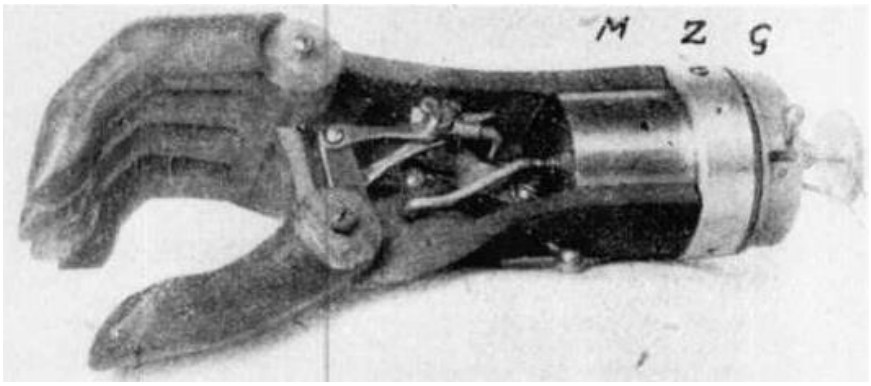


Fig. 2.6: Primera prótesis mioeléctrica [14]

Tras ella, la primera prótesis usada fue la llamada “Mano Rusa” [15]. Inicialmente, las señales mioeléctricas eran utilizadas para detectar la intención de movimiento y, así, activar la prótesis que, en general, realizaba un movimiento de pinza. Desde entonces, se han llevado a cabo numerosos estudios como el realizado por Hudgins [16], donde se inició el control mioeléctrico basado en reconocimiento de patrones, en concreto, con la aplicación de redes neuronales.

Estas prótesis han experimentado una gran evolución y, en la actualidad, se pueden encontrar algunas como la “Bebionic Hand” [17] o la “i-Limb Ultra” [18], las cuales, incluso, controlan la velocidad y la fuerza del agarre. Disponen de diferentes motores, cada uno de los cuáles se encarga del movimiento de un dedo.

Fig. 2.7: i-Limb Ultra [18]

Estas últimas son, en ocasiones, demasiado caras para los pacientes, ya que pueden costar incluso más de 30.000 euros. Por ello, en los últimos años se han desarrollado numerosas investigaciones en el área de la impresión 3D. Este es el caso de "Bionico Hand" [19], un proyecto que desarrolla prótesis mioeléctricas funcionales de bajo coste, realizadas mediante impresión 3D.

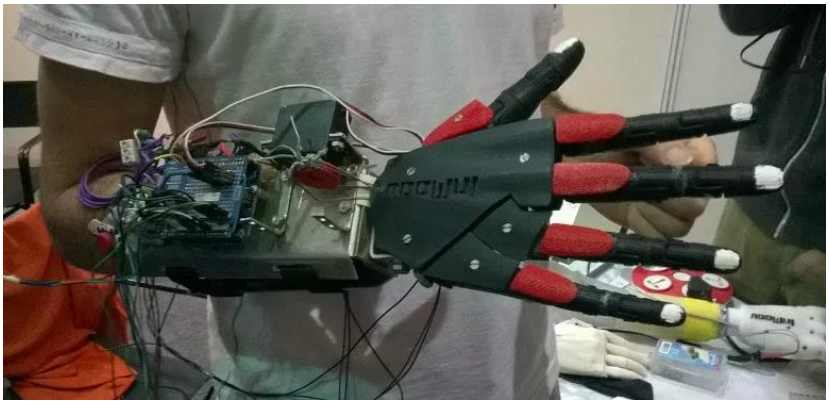


Fig. 2.8: Bionico Hand [19]

Los principales beneficios que aporta el uso de la impresión 3D, además de su bajo coste, generalmente inferior a los 300€, es la capacidad de adaptar el diseño a cada persona. Existen numerosos tipos de amputados, es decir, algunos pacientes necesitan prótesis completas de mano, de alguno de sus dedos o del brazo completo. Asimismo, el tamaño de las mismas varía. En el caso de los niños, que crecen de manera constante, el uso de este tipo de prótesis puede resultar especialmente beneficioso, ya que es más sencillo modificar las dimensiones del diseño.

Es por ello que, en la actualidad, se están realizando numerosas investigaciones en este campo ya que, a pesar de que las prótesis no tengan la misma precisión o comodidades que pueden ofrecer las anteriormente explicadas, sí se pueden obtener prótesis funcionales para utilizar en la vida cotidiana. Sin duda, la impresión 3D, puede representar el futuro de prótesis y exoesqueletos.

2.3. Prácticas docentes

Dada la finalidad docente del presente trabajo, en este apartado se pretenden mostrar otras prácticas realizadas basadas en la electromiografía.

En primer lugar, es importante hablar de las prácticas actualmente realizadas en la asignatura de "Aplicaciones de la automática en biomédica" para, posteriormente, poder situar en la asignatura la maqueta desarrollada en este trabajo.

La asignatura consta de dos prácticas, realizadas en varias sesiones. En la primera de ellas, denominada "Biomecánica del brazo humano" [20], el alumno aprende a modelar el bíceps en Simulink. En la segunda, "Procesamiento de señales de electromiografía" [21], mediante el uso de electrodos superficiales, los alumnos aprenden a procesar señales EMG para controlar un servomotor mediante el uso de un umbral. Como parte final de la asignatura, se propone al alumno el diseño de una interfaz gráfica en MATLAB en la que se pueda visualizar un juego controlado con las señales EMG adquiridas.

En otras universidades, existen titulaciones del campo Biomédico en las que se realiza el análisis de la señales EMG fuera del contexto del control de dispositivos. En el "Máster de Ingeniería Biomédica" de la Universidad Politécnica de Madrid, se realizan las prácticas "Análisis cuantitativo de actividad voluntaria rítmica y temblor" [22] y "Análisis cuantitativo del registro del reflejo rotuliano" [22], en la que se estudian las señales EMG, desde su origen en los músculos hasta su tratamiento, analizándolas en diferentes estados del paciente y sometiendo a los músculos a diferentes estímulos.

En el caso de Facultad de Medicina de la UNAM, existen prácticas de laboratorio de fisiología orientadas al estudio de la Electromiografía [23]. En ellas, se plantea un problema médico al alumno y debe resolverlo mediante la realización de un estudio electromiográfico.

3. Detección de movimiento en función de la señal EMG

3.1. Adquisición de señales EMG

3.1.1. Electromiografía

Antes de definir qué es la electromiografía, hay varios conceptos que se deben comprender, como la fisiología de los músculos. En primer lugar, hay que diferenciar los tres tipos de músculos: cardíaco, liso y esquelético. Siendo estos últimos los responsables del movimiento. El movimiento es debido a las unidades motoras, formada por "una neurona motora (…), su axón y las fibras musculares que inerva" [24]. La contracción muscular es debido a la acetilcolina, una sustancia liberada por la terminación de la neurona que, al unirse con los receptores de los músculos, se produce un flujo de iones, de manera que entra Na+ y Ca+ al interior y sale K-, creando el potencial de acción, el cual produce la contracción del músculo [25]. Las neuronas continuamente generan este potencial de acción de manera que, al medir sobre la superficie de la piel, se obtiene la señal mioeléctrica (también conocida como EMG o de electromiografía) como el sumatorio de todos los potenciales de acción. Estas señales presentan una amplitud entre -5mV y 5mV y una frecuencia entre 0 y 500 Hz [26].

Una vez se conoce la fisiología de los músculos, podemos definir electromiografía como la "adquisición, registro y análisis de la actividad eléctrica generada en nervios y músculos a través de la utilización de electrodos" [27], tal y como se muestra en la Fig.3.1.

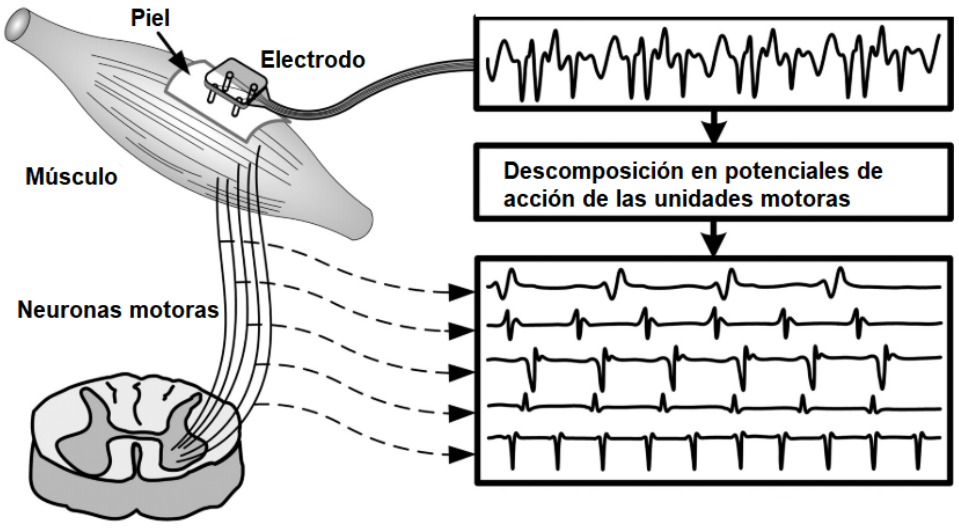


Fig. 3.1: Adquisición de señal EMG [28]

Existen dos tipos de señales EMG, intramuscular, medida con electrodos invasivos, y superficial, medida con electrodos no invasivos. Los electrodos invasivos o intramusculares consisten en una aguja que mide la señal directamente del músculo, lo que proporciona señales de alta calidad y, además, permite medir un solo potencial de acción. Por ello, este tipo de electrodos se utiliza, normalmente, en estudios médicos.

En cuanto a los electrodos no invasivos o superficiales, tal y como su nombre indica, se sitúan en la superficie de la piel. Estos cuentan con una superficie conductora, generalmente de Ag/AgCl. Debido a que proporcionan una mayor comodidad al paciente que los electrodos de aguja, son los más utilizados en el control de prótesis.

La principal desventaja de medir las señales EMG superficialmente es que, además de detectar la actividad eléctrica del músculo sobre el que están situados los electrodos, recogen la actividad de músculos cercanos y, a su vez, la señal recogida presenta ruido, por lo que es necesario procesar la señal para eliminarlo.

Además, un último factor a tener en cuenta es el contacto del electrodo con la piel, ya que el sudor, la fatiga o la temperatura influyen en la medida. Por ello, generalmente se aplica un gel conductor entre el electrodo y la piel de manera que el contacto eléctrico mejora y, así, se puedan adquirir señales de mayor calidad.

3.1.2. Myo Armband

El Myo Armband, mostrado en la Fig.3.2, es un dispositivo que, situado en el antebrazo del usuario, es capaz de leer las señales EMG de los músculos y, así, reconocer los distintos gestos que realiza el usuario (Fig.3.3)

Fig. 3.2: Myo Armband [29]

Esto es posible gracias a sus ocho electrodos, que registran la actividad con una frecuencia de 200Hz. Además, cuenta con un giroscopio, un acelerómetro y un magnetómetro, que permiten identificar la dirección y el movimiento del brazo [2].

Fig. 3.3: Gestos reconocidos por el Myo Armband [29]

3.1.2.1. Especificaciones técnicas

En cuanto a sus dimensiones, como se representa en la Fig.3.4, el Myo Armband puede ser utilizado por usuarios con una circunferencia del antebrazo entre los 19 y los 34 cm. Tiene un peso de 93 gramos y un grosor de 1.14 cm.

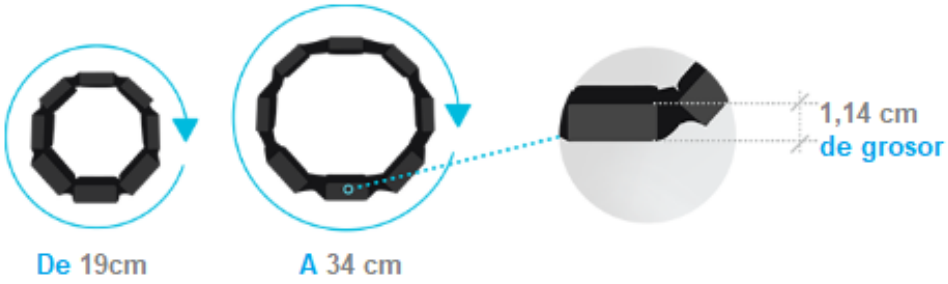


Fig. 3.4: Dimensiones del Myo Armband [29]

Los sensores EMG son de acero inoxidable de grado médico, posee un giroscopio de tres ejes, un acelerómetro de tres ejes y un magnetómetro de tres ejes. Además, contiene un procesador ARM Cortex M4.

Además, es compatible con numerosos dispositivos de diferentes sistemas operativos, como Windows, Mac, iOS y Android. La comunicación entre ellos es posible gracias a su tecnología Bluetooth®, lo cual permite ser utilizado de forma inalámbrica [29].

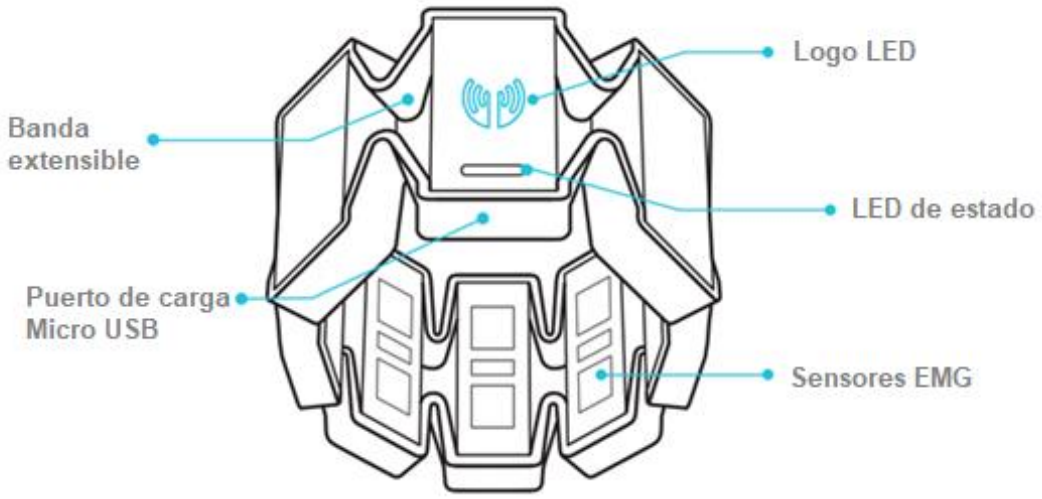


Fig. 3.5: Componentes principales del Myo Armband [30]

3.1.2.2. Aplicaciones

El Myo Armband cuenta con numerosas aplicaciones. Fundamentalmente, estas están orientadas al control de diferentes aplicaciones, como Netflix, YouTube o Spotify, así como al manejo de presentaciones en Prezi o PowerPoint.

Además, se utiliza para el control de videojuegos y drones y, en el campo médico, se utiliza en los quirófanos para facilitar al cirujano la visualización de imágenes médicas sin necesidad de tener contacto ni con el paciente ni con otro dispositivo, evitando así posibles infecciones y, también, permite reducir la duración de las operaciones.

Pero lo que hace realmente destacable a este dispositivo, es la posibilidad de modificar su software inicial para que los usuarios puedan registrar nuevos movimientos y configurarlo de la manera que deseen. Así, recientes estudios han logrado realizar traducciones de lengua de signos gracias al Myo Armband [31].

De aquí, nace la idea de aplicar este dispositivo en el área de la biomedicina, ya que permitiría realizar el control de prótesis y exoesqueletos gracias al estudio de las señales registradas tras la modificación del SDK (Software Development Kit).

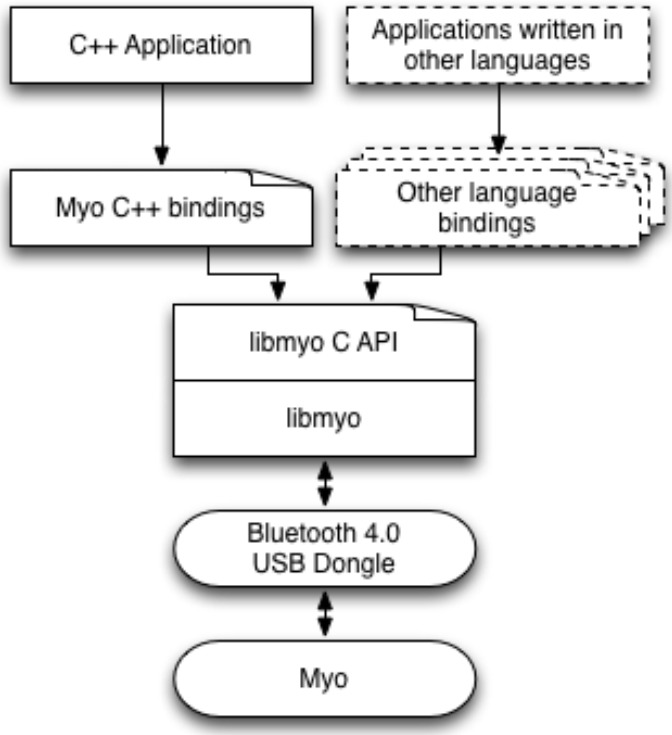


Fig. 3.6: Diagrama de funcionamiento del Myo Armband con otras aplicaciones [30]

3.1.3. Adquisición de datos

El primer paso para desarrollar el proyecto consiste en la visualización de los posibles movimientos que se pueden realizar y seleccionar los dos más diferenciados, con el objetivo de realizar una maqueta sencilla pero funcional. En esta ocasión, los movimientos seleccionados son:

- Movimiento 1: mano abierta
- Movimiento 2: mano cerrada

Una vez seleccionados los movimientos, se comienza con la adquisición de las señales de los diferentes movimientos con las que, posteriormente, se llevará a cabo la generación y entrenamiento de la red neuronal.

Esta adquisición se realiza mediante el Myo Armband, colocado en el antebrazo de la manera que se indica en la Fig.3.7.

Fig. 3.7: Myo Armband situado en el antebrazo

Como se ha mencionado anteriormente, se parte de un trabajo previo en el que se modificó el software inicial del Myo Armband para poder adquirir las señales. Por tanto, se comienza utilizando el programa *Quickstart_EMG_desagregados.m*, para realizar la toma de datos.

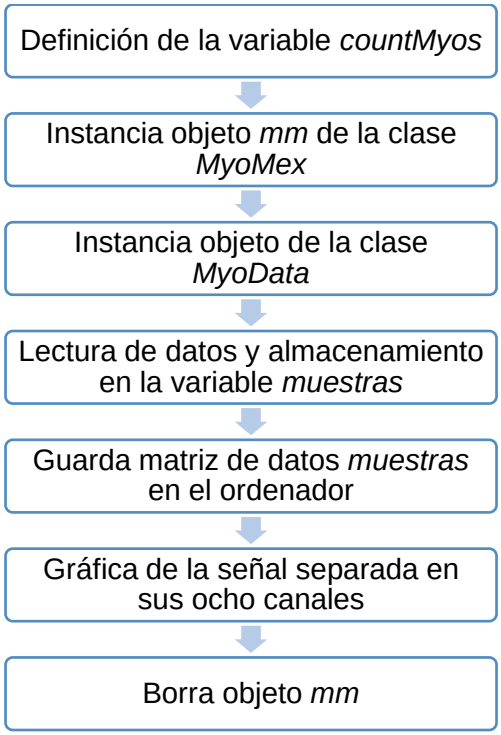


Fig. 3.8: Flujograma del código Quickstart_EMG_desagregados.m

En primer lugar, se indica el número de dispositivos Myo que se van a utilizar. En este caso es uno, pero en algunas ocasiones se pueden utilizar dos con el objetivo de mejorar la precisión en los datos adquiridos. Este valor se especifica en la variable *countMyos*, definida dentro del script, con la que se va a instanciar un objeto *mm* de la clase *MyoMex*.

A continuación, se guardan los datos que se van tomando en la variable *muestras,* definida dentro del script. Esta variable es una matriz que contiene tantas columnas como canales tiene el dispositivo, es decir, ocho. En cuanto a las filas, habrá tantas como muestras se hayan tomado. El número de muestras depende del tiempo de muestreo, el cual se indica dentro del script mediante el comando pause(*X*), siendo x el tiempo de muestreo deseado. Para la adquisición de datos de entrenamiento se usa un tiempo de muestreo de 5 segundos. El Myo Armband tiene una frecuencia de 200Hz, por lo que hay un total de 1000 muestras, es decir, 1000 filas. Obtenemos, por tanto, una matriz de dimensiones 1000x8.

	Canal 1	Canal 2	…	Canal 7	Canal 8
Muestra 1					
Muestra 2					
…					
Muestra 999					
Muestra 1000					

Tabla 3.1: Matriz adquisición de datos

Tras varias tomas, se observa que el número de muestras adquiridas (número de filas de la matriz anterior) varía. Posiblemente, debido a la pérdida de algún paquete Bluetooth en el envío. Por ello, posteriormente se recortarán las señales para que todas cuenten con el mismo número de muestras.

Una vez guardada la señal, se muestra la gráfica de la misma separada en los ocho canales con los que cuenta el Myo Armband, tal y como se muestra en las Fig.3.9 y 3.10.

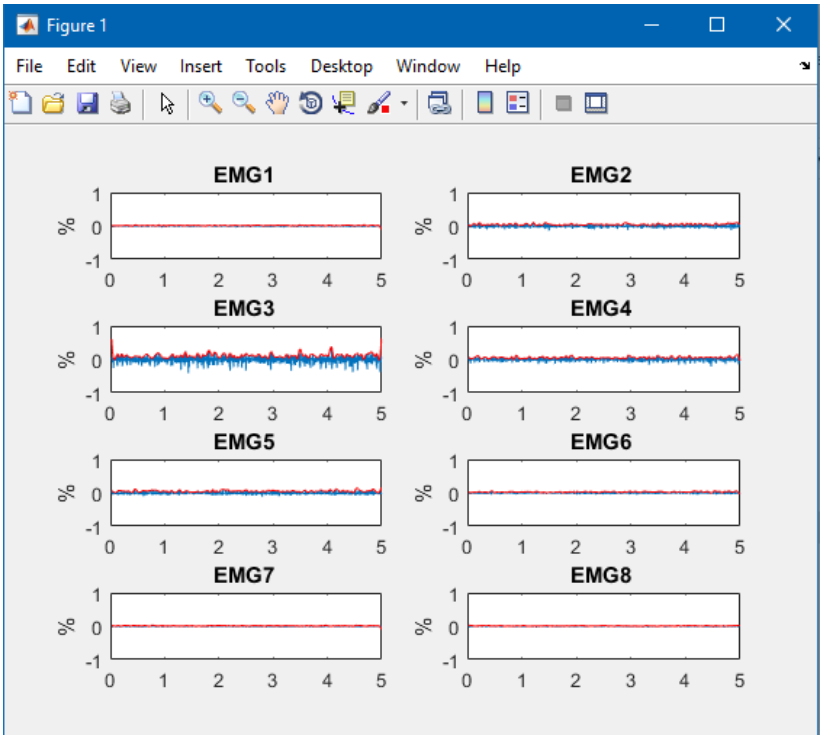


Fig. 3.9: Gráfica de señales EMG con mano abierta

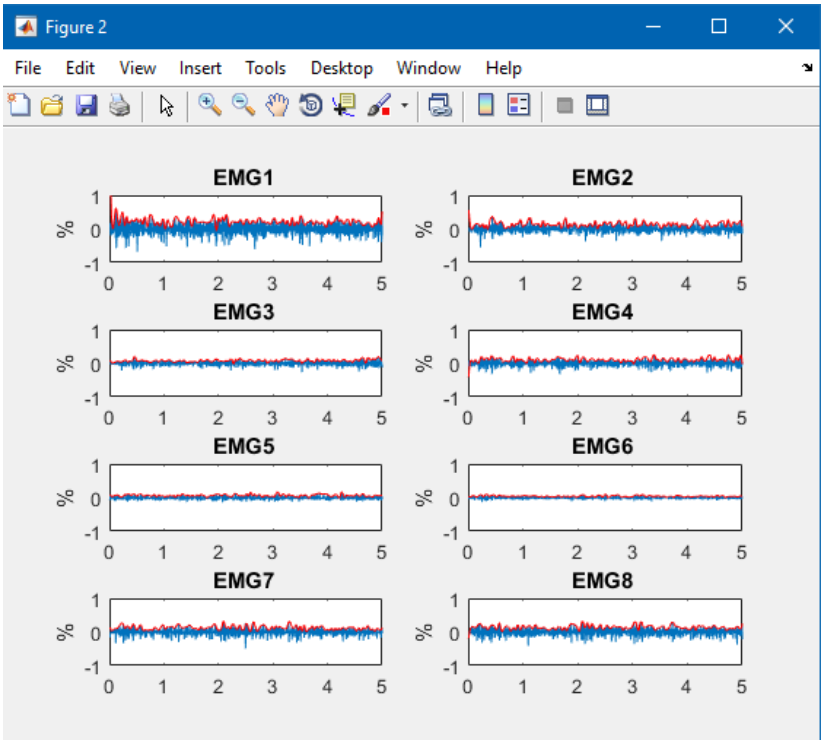


Fig. 3.10: Gráfica de señales EMG con mano cerrada

En ellas, se puede observar que con la mano abierta apenas hay actividad, mientras que con la mano cerrada, el sensor 1, situado sobre el músculo braquiorradial, presenta una gran actividad, puesto que es el músculo que podemos ver que más se contrae al cerrar el puño. Asimismo, ocurre con el canal 4, aunque en menor medida. Al obtener señales tan diferentes de los dos movimientos la red neuronal será capaz de identificarlos correctamente.

Para poder entrenar la red posteriormente y poder clasificar el movimiento entre abierto o cerrado, se toman 50 señales de cada uno de los movimientos con los nombres de *movX_Y*, siendo X el número de movimiento (1 abierto, 2 cerrado) e Y el número de la repetición.

Una vez realizada la toma de datos contamos con 100 señales (50 correspondientes al movimiento de apertura y 50 al de cierre), suficiente para entrenar la red neuronal.

3.2. Procesamiento de señales EMG

Antes de proceder con la red neuronal, es necesario procesar las señales, para ello se elaboran los códigos *CARGA_mov1* y *CARGA_mov2*. En el flujograma de la Fig. 3.11 se muestra el proceso llevado a cabo en este código, siendo X el nº de movimiento (1 abrir, 2 cerrar), Y el número de canal y Z el número de iteración.

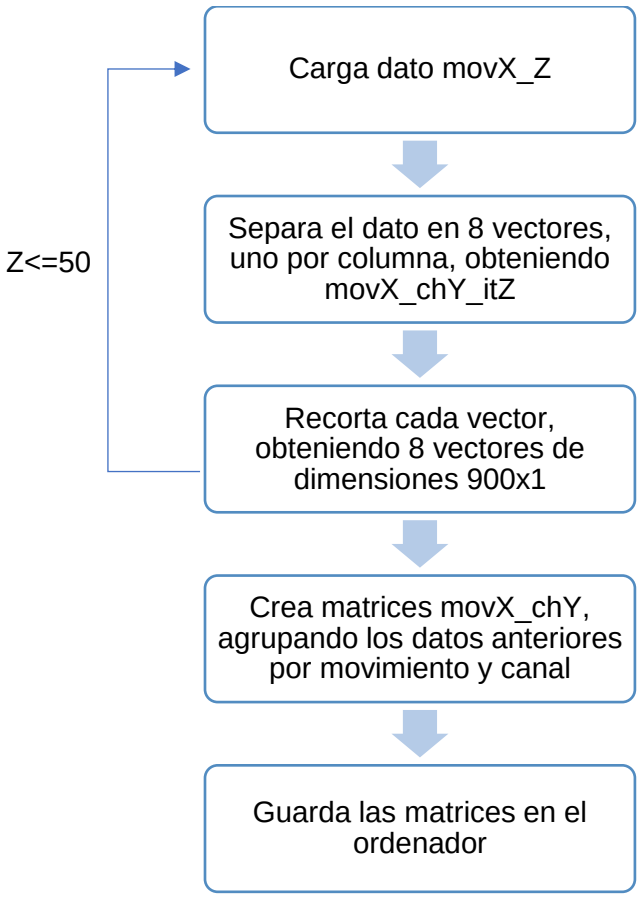


Fig. 3.11: Flujograma del código CARGA_MOVx

En primer lugar, en estos códigos se separa cada señal en los ocho canales y, a continuación, tal y como se ha dicho anteriormente, es necesario recortarla, ya que no todas las señales tienen 1000 muestras. Además, las primeras muestras que se toman de cada señal pueden corresponderse con las transiciones entre movimientos. Por ello, se eliminan las primeras 90 muestras y las posteriores a las 990. Se obtienen ahora matrices de dimensiones 900x1. Por último, hay que crear una matriz por movimiento que conste de todas las señales tomadas agrupadas por canal, como se muestra en la Tabla 3.2.

			Repetición 1	Repetición 2	...	Repetición 49	Repetición 50
Movimiento 1	Canal 1	Muestra 1					
		Muestra 2					
		...					
		Muestra 899					
		Muestra 900					

Tabla 3.2: Matriz de datos agrupados por canal y movimiento

Se obtienen, por tanto, matrices de dimensiones 900x50. Estas matrices se guardan con el nombre movX_chY, donde X es el número del movimiento (1 el de apertura y 2 el de cierre) e Y es el número del canal (del 1 al 8).

Se cuenta con un gran volumen de datos para entrenar la red. Es posible entrenarla con los datos en bruto pero tardará más tiempo en entrenarse y no será tan precisa. Por ello, primero es necesario procesar las señales. Este procedimiento se realiza con el código *time_domain_feature_extraction.m*, representado en el flujograma de la Fig.3.12.

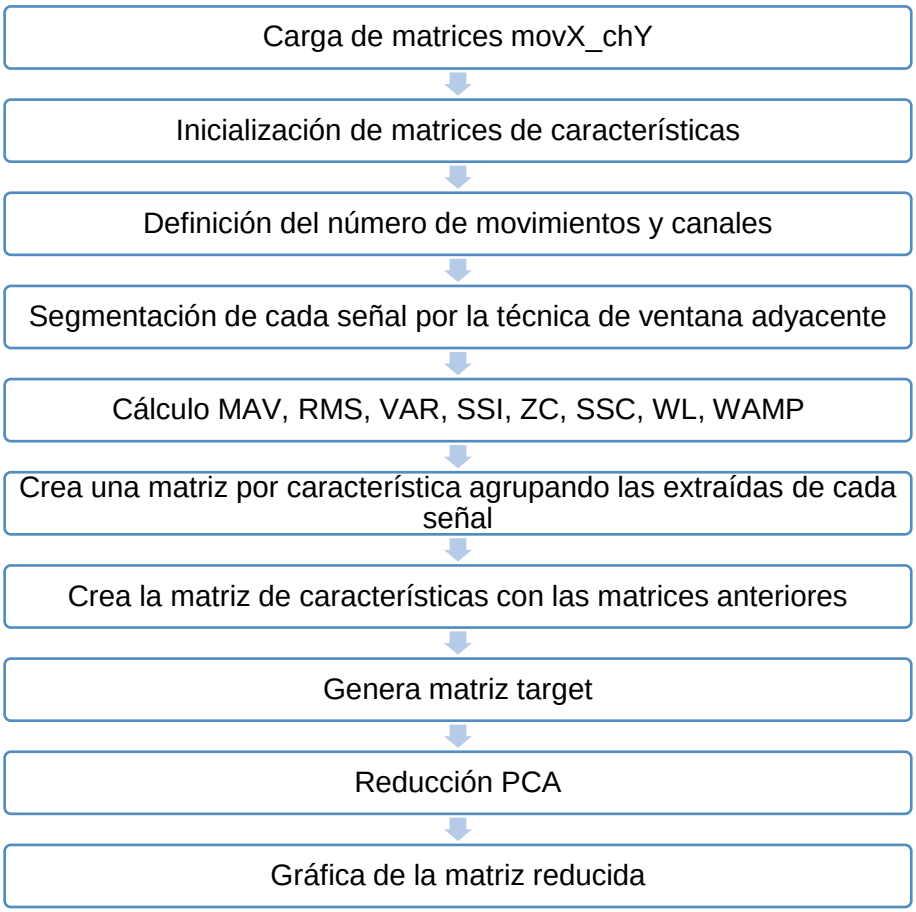


Fig. 3.12: Flujograma del código time_domain_feature_extraction.m

Este código, originalmente estaba preparado para realizar el procesamiento de las señales de cinco movimientos adquiridas a través de dos canales. Por ello, una vez comprendido su funcionamiento, se ha adaptado a los requerimientos del presente proyecto, siendo ahora capaz de procesar señales de dos movimientos adquiridas mediante ocho canales.

3.2.1. Segmentación

En primer lugar, se lleva a cabo la segmentación de la señal, para después extraer características de cada uno de los segmentos obtenidos y, así, poder determinar posteriormente de una manera más sencilla el tipo de movimiento realizado. La longitud de cada segmento tiene que ser inferior a los 300 ms y superior a los 200ms para tener el número suficiente de datos, según la literatura [32]. Por este motivo, la longitud de cada ventana es de 250ms, es decir, 50 muestras por segmento.

Tal y como se ha explicado anteriormente, existe un estado de transición entre dos movimientos (Fig. 3.13) en los que los datos se clasifican con menor precisión, a diferencia del estado estacionario. Por este motivo, con el objetivo de mejorar la clasificación se eliminan 90 muestras de cada señal.

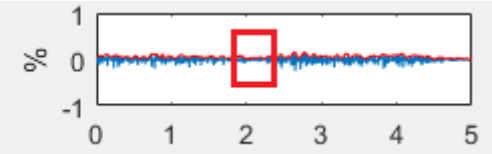


Fig. 3.13: Señal EMG con transición de movimiento

A continuación, hay que elegir la técnica de ventanas de datos: ventanas superpuestas, mostrada en la Fig. 3.14, o ventanas adyacentes de la Fig. 3.15. La primera técnica, consiste en el “desplazamiento de un segmento sobre otro con un incremento mayor que el tiempo de procesamiento y más corto que la longitud del segmento” [32] mientras que, la segunda, consiste en un segmento de una longitud fija.

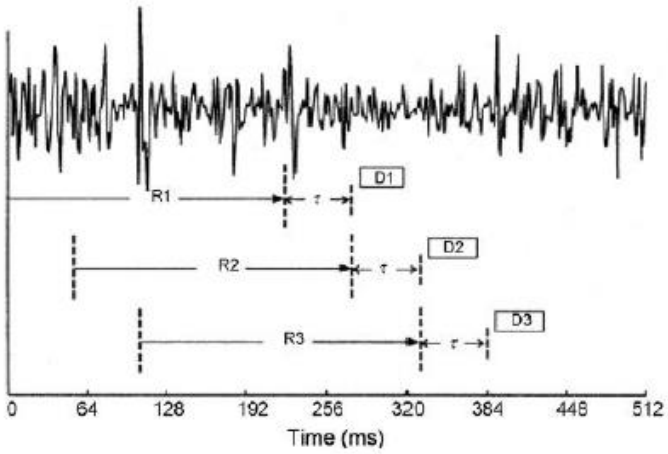


Fig. 3.14: Técnica de ventanas superpuestas [32]

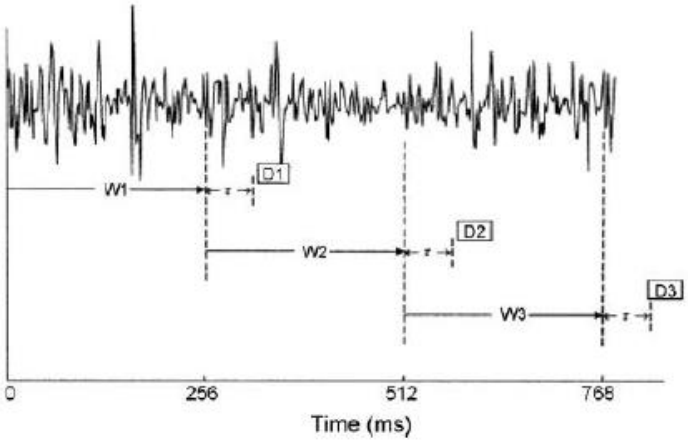


Fig. 3.15: Técnica de ventanas adyacentes [32]

En esta ocasión, se ha elegido el método de la ventana adyacente, ya que el método de las ventanas superpuestas incrementa el tiempo de procesamiento sin suponer una mejora para la extracción de características.

3.2.2. Extracción de características

Una vez segmentada la señal, se lleva a cabo la extracción de características. Esta puede ser analizada en el dominio del tiempo o de la frecuencia. Dado que en el dominio del tiempo se realiza de manera más rápida al no tener que realizar transformaciones a la señal, se tomarán las siguientes características de este dominio: Media del Valor Absoluto (MAV), Media Cuadrática (RMS), Varianza de la EMG (VAR), Integral Cuadrática Simple (SSI), Cruce por Cero (ZC), Longitud de la Forma de Onda (WL), Cambio de Pendiente de la Señal (SSC) y Amplitud de Willison (WAMP).

- Media del Valor Absoluto (MAV): es el valor medio absoluto de la señal. Es un indicador de la contracción muscular [34].

$$MAV_k = \frac{1}{N}\sum_{i=1}^{N}|x_i| \quad (1)$$

- Media cuadrática (RMS): representa la fuerza de la señal, de manera que es un indicador de la constante de fuerza [33]

$$RMS_k = \sqrt{\frac{1}{N}\sum_{i=1}^{N}{x_i}^2} \quad (2)$$

- Varianza de la EMG (VAR): mide la densidad de potencia que presenta la señal, siendo un indicador de la fuerza muscular [34].

$$VAR_k = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2 \quad (3)$$

- Integral Cuadrática Simple (SSI): mide la energía de la señal [33].

$$SSI_k = \sum_{i=1}^{N}(|x_i|)^2 \quad (4)$$

- Cruce por Cero (ZC): representa el número de veces que la señal pasa por cero. De esta manera, se puede calcular la frecuencia de la señal [34].

$$ZC\ aumenta\ si:$$
$$\{x_i > 0\ \ y\ \ x_{i+1} < 0\}\ \ o\ \ \{x_i < 0\ \ y\ \ x_{i+1} > 0\}\, y\, |xi - x_{i+1}| \geq \in \quad (5)$$

- Longitud de la Forma de Onda (WL): "es la longitud acumulada de la forma de onda sobre el segmento de tiempo" [33]. Aporta información de la amplitud de la señal, la frecuencia y el tiempo.

$$WL_k = \sum_{i=1}^{N-1}|x_{i+1} - x_i| \quad (6)$$

- Cambio de pendiente de la Señal (SSC): permite conocer, al igual que ZC, la frecuencia de la señal.

$$SSC\ aumenta\ si: \\ \{x_i > x_{i-1}\ y\ x_i > x_{i+1}\}\ o \\ \{x_i < x_{i-1}\ y\ x_i < x_{i+1}\}\ y\ |x_i - x_{i+1}| \geq \in\ o\ |x_i - x_{i-1}| \geq \in \quad (7)$$

- Amplitud de Willison (WAMP): es el número de veces que la diferencia de la amplitud entre puntos de datos adyacentes excede un umbral, siendo un indicador del nivel de contracción del músculo [34].

$$WAMP_k = \sum_{i=1}^{N-1} f(|x_i - x_{i+1}|) \quad (8)$$

$$f(x) = \begin{cases} 1, x > \in \\ 0, otro\ caso \end{cases}$$

Siendo k el segmento, x_i la longitud de cada parte del segmento, N la longitud del segmento, $\bar{x}$ el valor medio del segmento k y Є un umbral.

Una vez extraídas las características, tenemos una matriz de características (*features*) de dimensiones mxn, de manera que:

$$m = nºmovimientos * nºiteraciones * nºsegmentos \quad (9)$$

$$n = nºcanales * nºcaracterísticas \quad (10)$$

Sustituyendo en (9) y (10), obtenemos:

$$m = 2 * 50 * \left(\frac{900}{50}\right) = 1800$$

$$n = 8 * 8 = 64$$

Por tanto, en esta ocasión, tenemos una matriz de características de dimensiones 1800x64, mostrada en la Tabla 3.3.

			Característica 1			...	Característica 8		
			Canal 1	...	Canal 8		Canal 1	...	Canal 8
Movimiento 1	Iteración 1	Segmento 1							
		...							
		Segmento 18							
	...	...							
	Iteración 50	Segmento 1							
		...							
		Segmento 18							
Movimiento 2	Iteración 1	Segmento 1							
		...							
		Segmento 18							
	...	...							
	Iteración 50	Segmento 1							
		...							
		Segmento 18							

Tabla 3.3: Matriz de características

Además, una vez extraídas las características, en este mismo código se crea la matriz objetivo (*target*) que, posteriormente, se usará al generar la red neuronal. Esta matriz está formada por los números 0 y 1, que representan cada uno de los gestos que se han de reconocer.

3.2.3. Reducción de dimensionalidad

El último paso antes de proceder con la clasificación, es la reducción de dimensionalidad. A pesar de que mediante la extracción de características se ha reducido considerablemente el número de datos, todavía no es suficiente para entrenar el clasificador. Hoy en día existen clasificadores que permiten este gran volumen de datos pero, como ya se ha mencionado anteriormente, la precisión en el reconocimiento del gesto será mayor si se reducen. Esto, se puede realizar mediante la selección de características o mediante la proyección de las mismas.

Para llevar a cabo la técnica de selección de características es necesaria la participación del humano, de manera que según su criterio elija las características a tener en cuenta. Por este motivo, se elige la proyección de características. Así, este procedimiento se puede realizar automáticamente y, además, permite extraer los mejores datos puesto que no se eliminan características que podrían aportar gran información para la clasificación.

Dentro de la proyección de características existen diversos métodos. Entre ellos, se encuentran LDA (Linear Discriminant Analysis) y PCA (Principal Component Analysis). El primero de ellos, Análisis de Discriminador Lineal, es un método estadístico de aprendizaje supervisado, en el que se generan atributos que buscan aumentar al máximo la driscriminatividad, es decir, busca la mayor separabilidad de las clases de manera lineal. La segunda técnica, Análisis de Componentes Principales, es un método de aprendizaje no supervisado que "encuentra un conjunto de bases que maximizan la varianza de los datos originales y que son ortogonales entre sí" [35].

Para este proyecto, se utiliza el método de PCA, ya que es el más utilizado con las señales EMG dada su efectividad, dado que al ser señales de dinámica estocástica (presenta componentes aleatorias), el método LDA presenta menor efectividad.

Por tanto, con el método PCA, se reduce la matriz de características a dos dimensiones. De esta manera, se puede representar gráficamente, obteniendo el gráfico de la Fig.3.16.

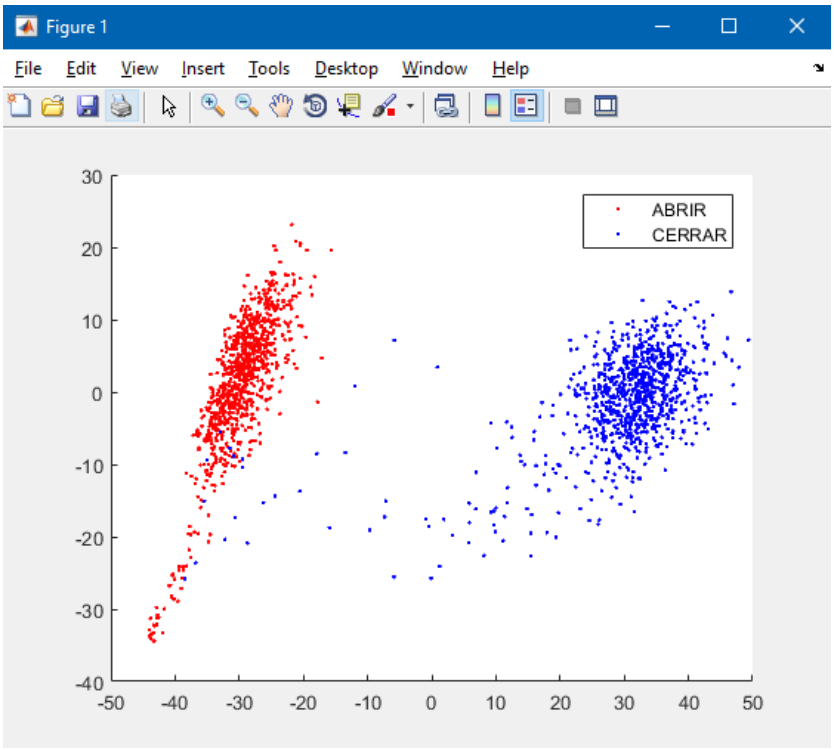


Fig. 3.16: Gráfica de la matriz de características tras la reducción PCA

Como se observa en él, los dos gestos están muy diferenciados, por lo que se espera una buena respuesta a la hora de realizar la clasificación. Por tanto, una vez realizada la reducción de dimensiones, ya se puede proceder a clasificar los gestos.

3.3. Clasificación de gestos

Una vez se han procesado las señales, ya se puede llevar a cabo la clasificación. Para ello, hay que tener en cuenta que las señales EMG pueden variar en gran medida en función de la persona de la que se están extrayendo, de la fatiga muscular de la misma e incluso de la posición y cantidad de electrodos. Por este motivo, el clasificador debe funcionar de manera adecuada a pesar de las variaciones en las señales, además de realizar la clasificación en el menor tiempo posible. Existen dos tipos de técnicas: técnicas no basadas en reconocimiento patrones y técnicas basadas en reconocimiento de patrones.

3.3.1. Técnicas no basadas en reconocimiento de patrones

El control sin reconocimiento de patrones normalmente se utiliza con prótesis sencillas, que no requieren la necesidad de identificar numerosos tipos de gestos. Dentro de este grupo se encuentran el control proporcional, el uso de umbrales y la máquina de estado finito [32].

3.3.1.1. Control proporcional

El control proporcional utiliza la amplitud de la señal EMG para controlar la velocidad o la fuerza del actuador situado en el miembro, prótesis o exoesqueleto, de la manera que se muestra en la Fig. 3.17.

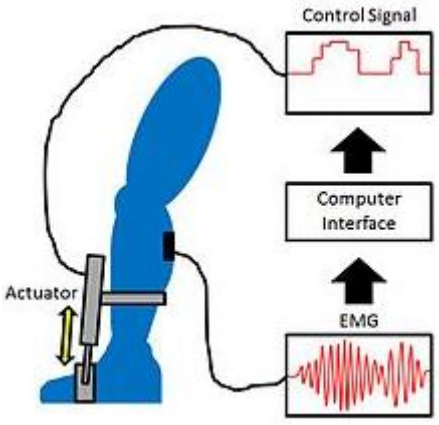


Fig. 3.17: Control proporcional aplicado a un actuador situado en el tobillo [36]

3.3.1.2. Umbrales

El control mediante umbrales consiste comparar la señal con diferentes umbrales indicados previamente. Pero, normalmente, se utiliza con un único umbral de manera que sirva para clasificar dos gestos diferentes. Cuando la señal está por encima del umbral, se realiza un movimiento, por ejemplo, el del cierre de la mano, y cuando está por debajo se mantiene abierta, tal y como se muestra en la Fig. 3.18.

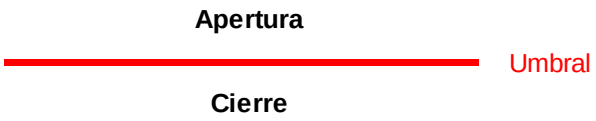


Fig. 3.18: Ejemplo de control de una prótesis de mano con umbral

También se pueden utilizar dos umbrales, de manera que se puedan realizar tres movimientos diferentes de la misma manera que se ha indicado anteriormente. Un ejemplo podría ser el control de una mano con tres estados: reposo, apertura y cierre, mostrado en la Fig. 3.19.

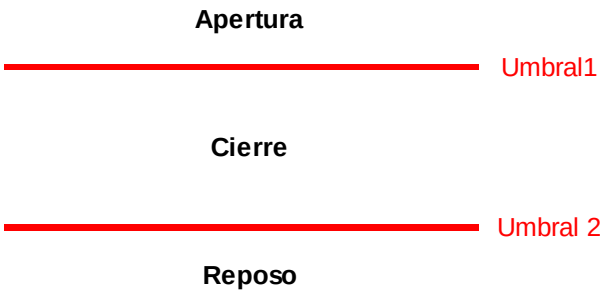


Fig. 3.19: Ejemplo de control de una prótesis de mano con dos umbrales

3.3.1.3. Máquina de estados finitos

La máquina de estados finitos consiste en diferentes movimientos que se pueden realizar según una secuencia prefijada. En otras palabras, existen diferentes estados y, el cambio de uno a otro se realiza aplicando un umbral como se ha explicado en el apartado 3.3.1.2 pero, en esta ocasión, este último estado se almacena, de manera que en función del siguiente movimiento se pasa a un estado u otro.

Un ejemplo, es la máquina de estados realizada por Felzer y Freisleben [37], mediante la cual se puede controlar una silla de ruedas. Esta máquina cuenta con los siguientes estados: recto, derecha, izquierda y detención. Y existen dos tipos de salidas de cada

uno de ellos: la primera (e1), que consiste en sobrepasar un umbral definido y, la segunda (e2) que consiste en sobrepasar este mismo umbral dos veces en un corto espacio de tiempo, tal y como se muestra en la Fig. 3.20. El funcionamiento de esta máquina de estados se muestra en la Fig. 3.21.

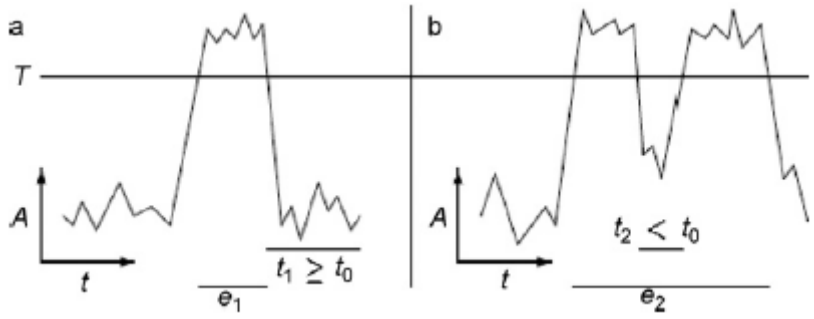


Fig. 3.20: Umbrales para el control de una silla de ruedas [32]

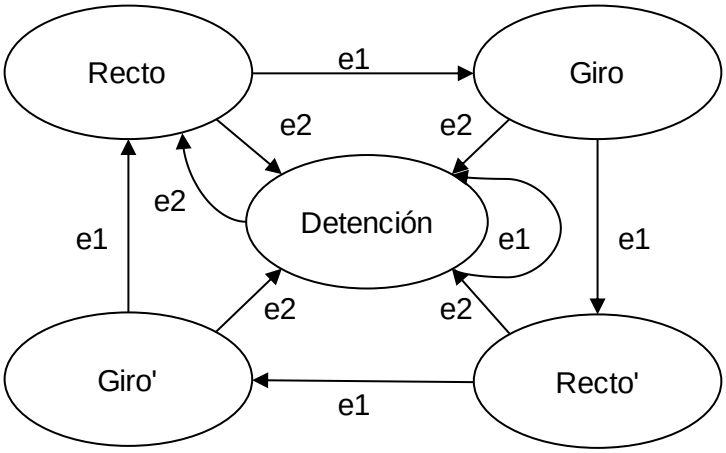


Fig. 3.21: Diagrama de transición de estados del control de una silla de ruedas [32]

3.3.2. Técnicas basadas en reconocimiento de patrones

Las técnicas basadas en el reconocimiento de patrones, son técnicas que utilizan unas determinadas características extraídas de las señales para la toma de decisión. Existen varios clasificadores basados en la técnica de reconocimiento de patrones. Entre los que destacan: Clasificador Bayesiano, Clasificador basado en la Lógica Fuzzy, Máquina de soporte vectorial y Redes Neuronales.

3.3.2.1. Clasificador Bayesiano

El método estadístico más utilizado es el clasificador bayesiano. Utiliza la probabilidad a priori para calcular la probabilidad a posteriori y, así, poder determinar a qué clase pertenece un dato. Este se basa en el Teorema de Bayes:

$$P(h/D) = \frac{P(D/h)P(h)}{P(D)}$$

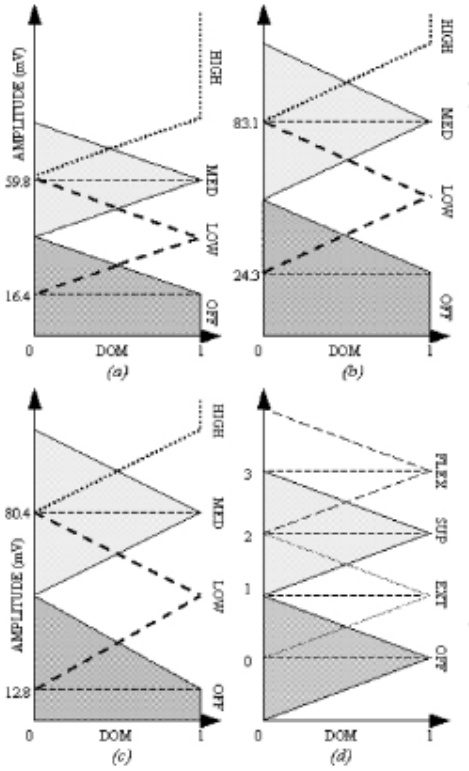
Siendo P(D/h) la probabilidad de observar los datos D dada la hipótesis h; P(h) es la probabilidad a priori de la hipótesis, mientras que P(D) es la probabilidad a priori de los datos. Obteniendo como resultado P(h/D), que es la probabilidad a posteriori de que se dé la hipótesis h según los datos D [38].

3.3.2.2. Lógica fuzzy

La lógica Fuzzy consiste en aplicar una serie de reglas de decisión, basadas en sistemas lineales, que permiten realizar aproximaciones no lineales [39]. Es decir, trata de simular el razonamiento humano en el que no todo es 0 o 1, sino que hay un rango de valores intermedios.

Una de las ventajas de utilizar este tipo de lógica en la clasificación de señales EMG es que, aun siendo señales de naturaleza estocástica, la lógica Fuzzy es capaz de encontrar patrones en ellas [32].

Un ejemplo de aplicación de la Lógica Fuzzy a señales EMG es el estudio realizado por Ajiboye y Weir [40] para una prótesis de brazo, en el que se obtuvo de un 94% a un 99% de éxito en la clasificación. Este consiste en la definición de las funciones *Off*, *Low*, *Med*, *High* (grado de la señal), de manera que, en función de ello se produzca una de las siguientes salidas: *Off*, *Extension*, *Supination* o *Flexion*. En la Fig 3.22 se muestran tantos las funciones como las reglas de decisión y la salida correspondiente.



IF			THEN
EMG Input 1	EMG Input 2	EMG Input 3	Output (User's Intent) is…
Off	Off	Off	Off
Low	Off	Off	Extension
Med	Off	—	Extension
Med	Low	—	Extension
Med	Med	—	Extension
High	—	—	Extension
Off	Low	Off	Supination
Off	Med	Off	Supination
Off	Med	Low	Supination
Low	Med	Off	Supination
Low	Med	Low	Supination
Off	High	—	Supination
Low	High	—	Supination
Off	Off	Low	Flexion
Off	Off	Med	Flexion
Off	Low	Med	Flexion
Low	Off	Med	Flexion
Low	Low	Med	Flexion
—	Med	Med	Flexion
—	—	High	Flexion

Fig. 3.22: Funciones de pertenencia y reglas del estudio Ajiboye y Weir [40]

3.3.2.3. Máquina de soporte vectorial

Una Máquina de soporte vectorial o, en inglés, SVM (*Support Vector Machine*) consiste en encontrar el hiperplano que separa en dos un conjunto de datos a la máxima distancia, es decir, separa los datos en las dos clases diferentes que existen, tal y como se muestra en la Fig. 3.23. Los datos que se encuentran más cercanos al hiperplano son denominados Vectores de Soporte o *Support Vectors* [41].

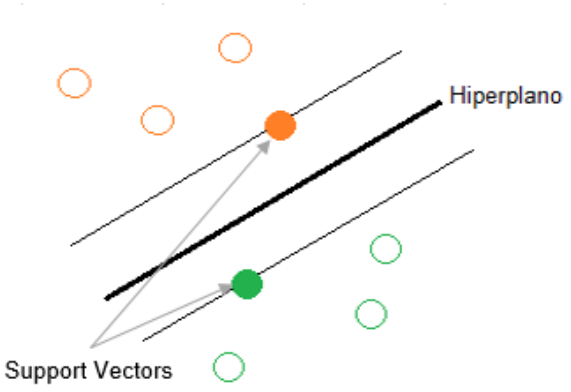


Fig. 3.23: Ejemplo gráfico de un SVM

El estudio realizado por Oskoei y Hu [42] para clasificar movimientos de la extremidad superior utilizando SVM en señales EMG, obtuvo un éxito en la clasificación de un 95.5%.

3.3.2.4. Redes neuronales

Desde el punto de vista biológico, las agrupaciones de neuronas interconectadas se denominan redes neuronales. Siguiendo el funcionamiento de estas, surgen las redes neuronales artificiales.

La representación matemática de una neurona se puede expresar de la siguiente manera:

$$neurona = \sum_{i=0}^{n} w_i x_i \tag{11}$$

Donde x representa las entradas y w los pesos.

A las neuronas se les aplica una transformación no lineal, denominada función de activación, obteniendo tras ello la salida, tal y como se muestra en el esquema de la Fig.3.24.

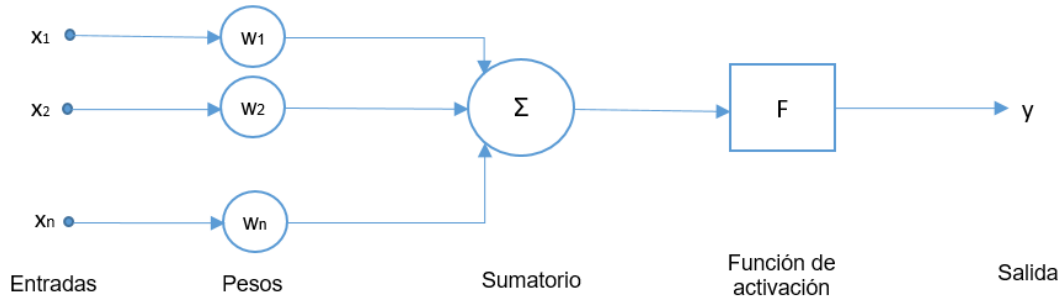


Fig. 3.24: Representación gráfica de una neurona

En cuanto a las redes neuronales, están compuestas por una capa de entrada, una o más capas ocultas y una capa de salida. Las señales se envían de una capa a la siguiente en el caso de conexiones *feedforward* (Fig. 3.25) o bien, se envían de la salida de una capa posterior a la anterior como es el caso de las conexiones *feedback* (Fig.3.26) [43].

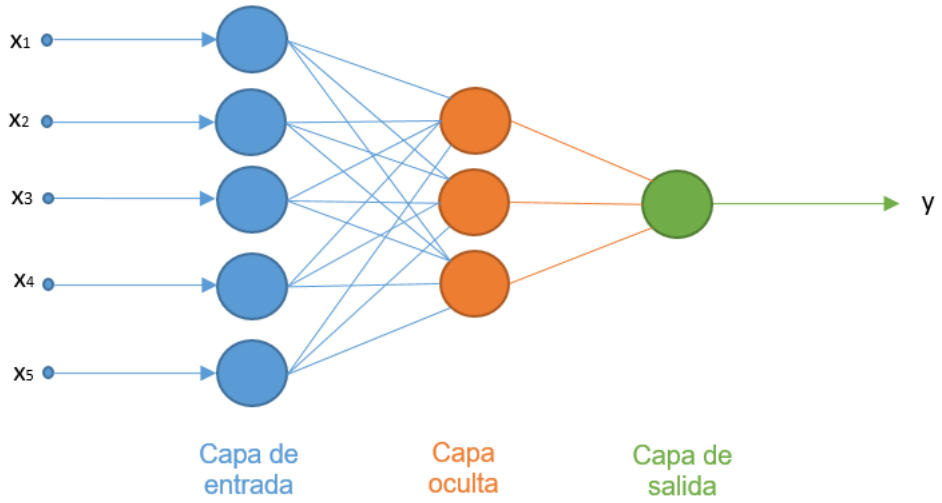


Fig. 3.25: Red neuronal feedforward

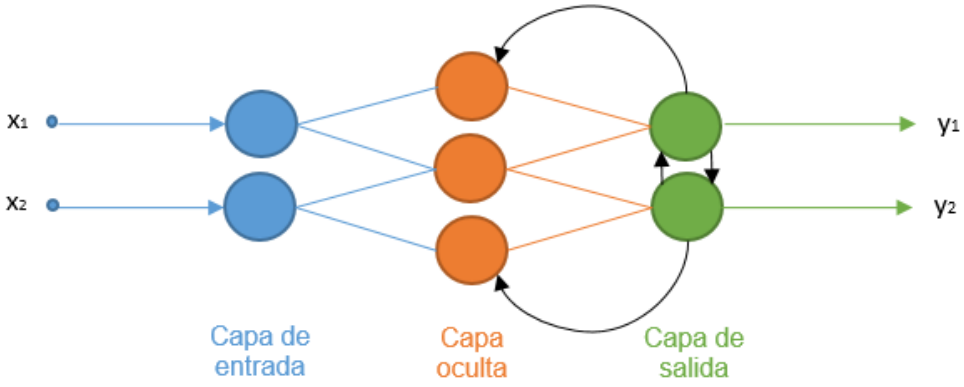


Fig. 3.26: Red neuronal feedback

La mayor ventaja que presenta el uso de este clasificador es la capacidad de adaptar sus parámetros a tiempo real [44]. Por este motivo, existe un uso extendido de las redes neuronales para la clasificación de señales biomédicas.

Un ejemplo de ello, es el estudio como parte del proyecto “Hand of Hope” [45]. En él, se desarrollan prótesis mioeléctricas de bajo coste, utilizando una red con conexión *feedforward* para el reconocimiento de patrones.

3.3.3. Implementación del clasificador para la maqueta

Para este proyecto, ambas técnicas, basadas en reconocimiento de patrones o no, son válidas, puesto que únicamente se van a realizar los movimientos de apertura y cierre de la mano pero, con el objetivo de realizar trabajos futuros en los que se detecte un rango más amplio de movimientos, se utiliza la técnica de reconocimiento de patrones.

Tras analizar los diferentes clasificadores que se pueden utilizar para la maqueta, se considera que tanto la red neuronal como el SVM son adecuados ya que son los clasificadores que mejor funcionan con señales de electromiografía. Siguiendo el criterio de utilizar recursos con los alumnos estén más familiarizados, se escoge la red neuronal como clasificador para el presente proyecto, ya que en asignaturas previas del grado como Control Inteligente se estudian en profundidad.

En primer lugar, es necesario generar y entrenar la red. Este proceso se lleva a cabo en el programa *TRAIN_NN.m,* mostrado en la Fig.3.27. El algoritmo consiste en la carga de las matrices *feat_pca* y *target* obtenidas tras la extracción de características y la reducción de dimensionalidad. Una vez cargadas en el programa, se genera la red mediante el comando *feedforwardnet.* Esta se guarda en una variable denominada *net.* Una vez generada, se entrena con las dos matrices cargadas utilizando la siguiente sintaxis: *train (net,feat_pca,target)*.

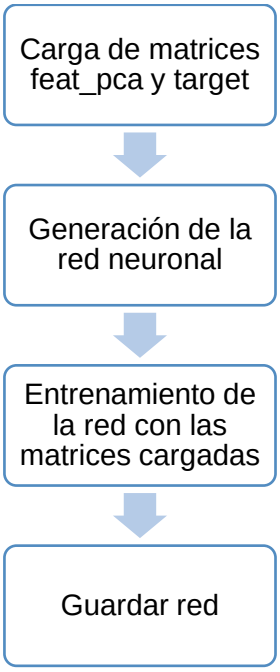


Fig. 3.27: Flujograma del programa TRAIN_NN.m

En la Fig. 3.28, se puede observar la red obtenida tras el entrenamiento, donde la entrada se corresponde con la matriz de características reducida y la salida con los dos

posibles movimientos. El tamaño de las capas ocultas lo puede definir el usuario en el código. En este caso, tras realizar pruebas con diferentes valores, se toma el valor 10.

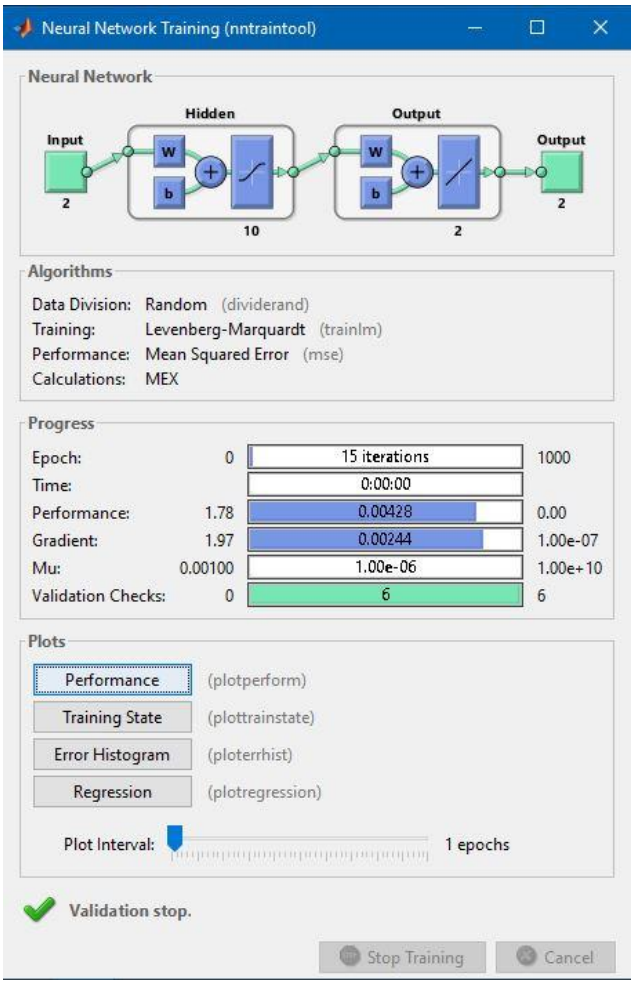


Fig. 3.28: Ventana de entrenamiento de la red neuronal

Una vez entrenada la red, ya está lista para clasificar nuevos datos. Posteriormente, en el capítulo 4 se explicará el código donde se realiza todo el proceso para una nueva señal adquirida y, por tanto, cómo funciona la simulación de la red neuronal que se ha entrenado.

Cabe destacar el error encontrado al tratar de usar la red neuronal de forma gráfica para realizar la integración de todas las partes en Simulink. Hasta la fecha, no se ha integrado el Myo Armband en Simulink, por lo que la adquisición de datos debía realizarse en MATLAB. El problema apareció al tratar de enviar los datos adquiridos a Simulink para, desde ahí, ya poder procesar la señal, clasificar el movimiento y enviar la señal a la maqueta. No era posible pasar de MATLAB a Simulink todos los datos adquiridos ya que demoraba varios minutos sin llegar a transferirse con éxito y, así, hubiera sido imposible lograr que la maqueta funcionara a tiempo real. Por este motivo, se decidió generar la red y entrenarla a través de código en MATLAB.

4. Funcionamiento a tiempo real

En la Introducción del proyecto, se señaló como objetivo el funcionamiento de la maqueta en tiempo real. Así, además de ser útil para los alumnos, sienta las bases para el futuro uso del Myo Armband en prótesis mioeléctricas.

Por ello, hay que integrar el sistema completo, para lo que se procede a crear el código *MAIN.m*, el cual será el que se ejecute para poner en funcionamiento la maqueta, tal y como se muestra en la Fig.4.1.

En este código, se cuenta con un bucle que se encarga de realizar todos los procesos que se han explicado en el capítulo anterior de manera continuada, permitiendo así adquirir datos, procesarlos, clasificar el gesto realizado y enviar la señal al servomotor en tiempo real.

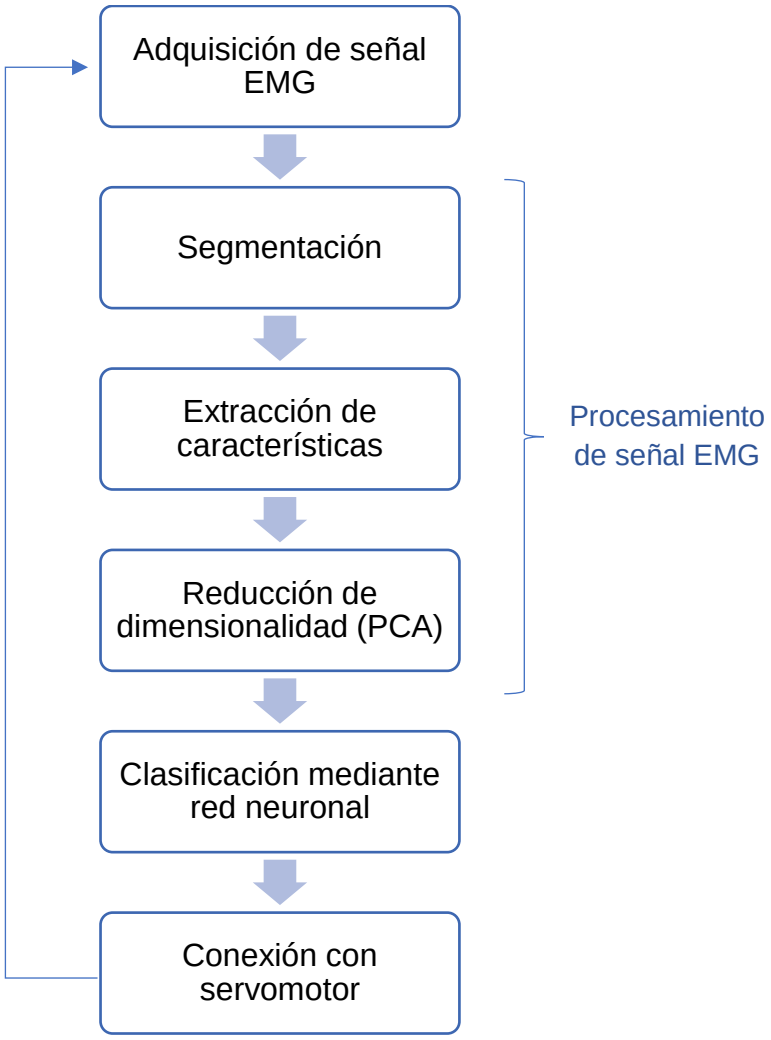


Fig. 4.1: Pasos del funcionamiento de la maqueta

En primer lugar, se define en el código una variable denominada *bucle,* la cual nos va a permitir elegir el número de veces que queremos que este se repita. Una vez estamos dentro del bucle, se procede a la toma de datos.

Para adquirir las señales que se usarían para entrenar la red neuronal, la toma de datos se realizaba durante 5 segundos pero, ahora, el objetivo es minimizar los tiempos para lograr el funcionamiento a tiempo real. Por ello, tras realizar varias pruebas, se fija un tiempo de lectura de datos de 0.5 segundos, debido a que con un tiempo inferior no había datos suficientes para reconocer correctamente el gesto que se estaba realizando.

Siguiendo los mismos pasos, una vez tomada la señal, se procesa. Dado que ahora solo hay una señal, directamente se separa en ocho vectores, correspondientes a los ocho canales, obteniendo vectores de alrededor de 100 elementos (*EMG_ch1, EMG_ch2*…). De nuevo, para eliminar posibles errores en la clasificación por el estado de transición, se eliminan las primeras 10 muestras y, para obtener señales de la misma longitud, se eliminan todas las muestras posteriores a la 90.

Una vez tenemos los datos separados por canales y con la misma longitud, se procede a segmentar. Tras segmentar, se extraen características de la misma manera que se explica en el capítulo 3 para, así, obtener una matriz de características. En esta ocasión, es importante tener en cuenta que la reducción de dimensionalidad PCA se realiza aplicando los mismos coeficientes que la primera vez que se llevó a cabo. Por ello, esta primera vez se guardaron en la variable *coeff*, la cual volverá a ser utilizada ahora.

Tras procesar las señales, se simula la red neuronal que se entrenó previamente, pasándole la matriz de características reducida que acabamos de obtener. Obteniendo así la matriz *resultado*, la cual se va a utilizar para poder determinar qué movimiento se ha realizado ya que tendrá un aspecto similar al de la matriz *target* con la que se entrenó la red, formada por 1 y 0. En esta ocasión, se obtendrá una matriz *resultado* en la que cada una de sus dos filas tendrá valores muy cercanos al 1 o muy cercanos al 0. Gracias a esto, podemos identificar el gesto, siguiendo el siguiente criterio: si la fila 1 de la matriz resultado es mayor que la segunda fila, el gesto se clasifica como “abierto”, en caso contrario, se identificad como “cerrado”.

Por pantalla se muestra la clasificación, tal y como se muestra en la Fig. 4.2, pero, además, existe una variable *movimiento* que toma el valor 0 cuando es “abierto” y 1 cuando es “cerrado”. Esta variable permite llevar a cabo el último paso: enviar la señal al servomotor.

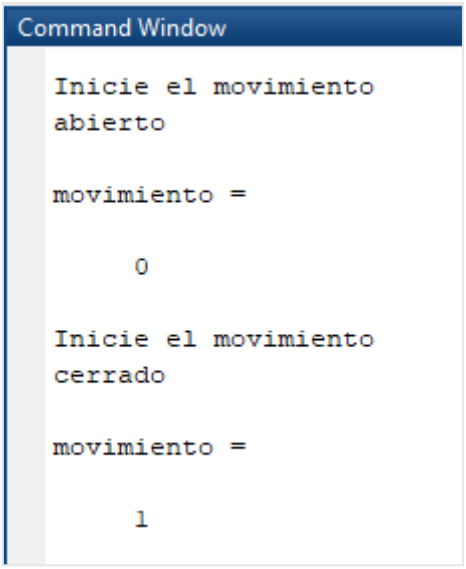


Fig. 4.2: Clasificación de movimiento en la ventana de comandos

4.1. Microcontrolador

En primer lugar, es necesario programar el microcontrolador para, después, enviar la señal PWM (Pulse Width Modulation) al servomotor. El microcontrolador sirve de intermediario entre el ordenador y el servomotor. En este caso, se ha elegido el modelo *STM32F4 Discovery,* mostrado en la Fig.4.3, dado que presenta un coste inferior a otros microcontroladores similares, tales como Arduino Uno o Raspberry Pi, además de tener buenas prestaciones, adecuadas para este proyecto, como la posibilidad de programar salidas PWM.

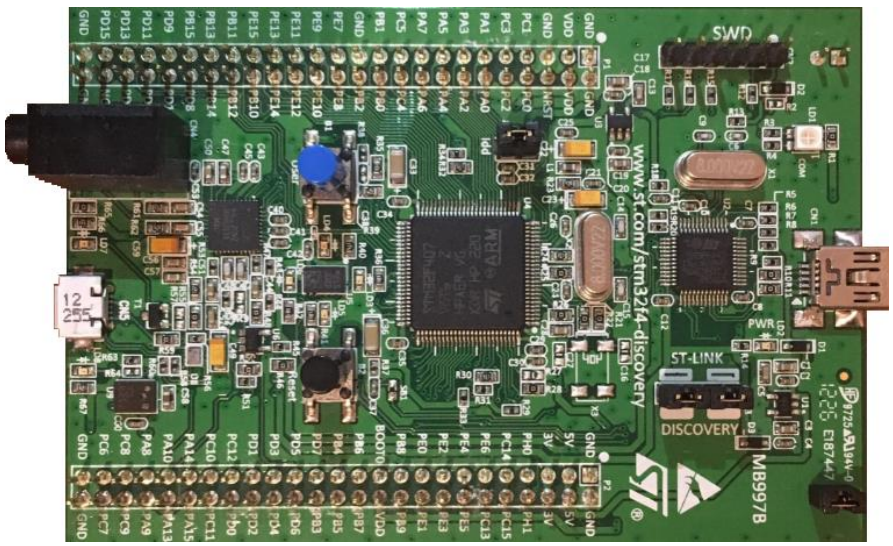


Fig. 4.3: Microcontrolador STM32F4 Discovery

STM32F4 Discovery	Arduino Uno	Raspberry Pi
16,83 €	20,00 €	36,00 €

Tabla 4.1: Comparativa de precios de microcontroladores

4.1.1. Programación del microcontrolador

La programación del microcontrolador se ha llevado a cabo mediante un modelo de Simulink, *servo_usb.mdl,* mostrado en la Fig.4.4.

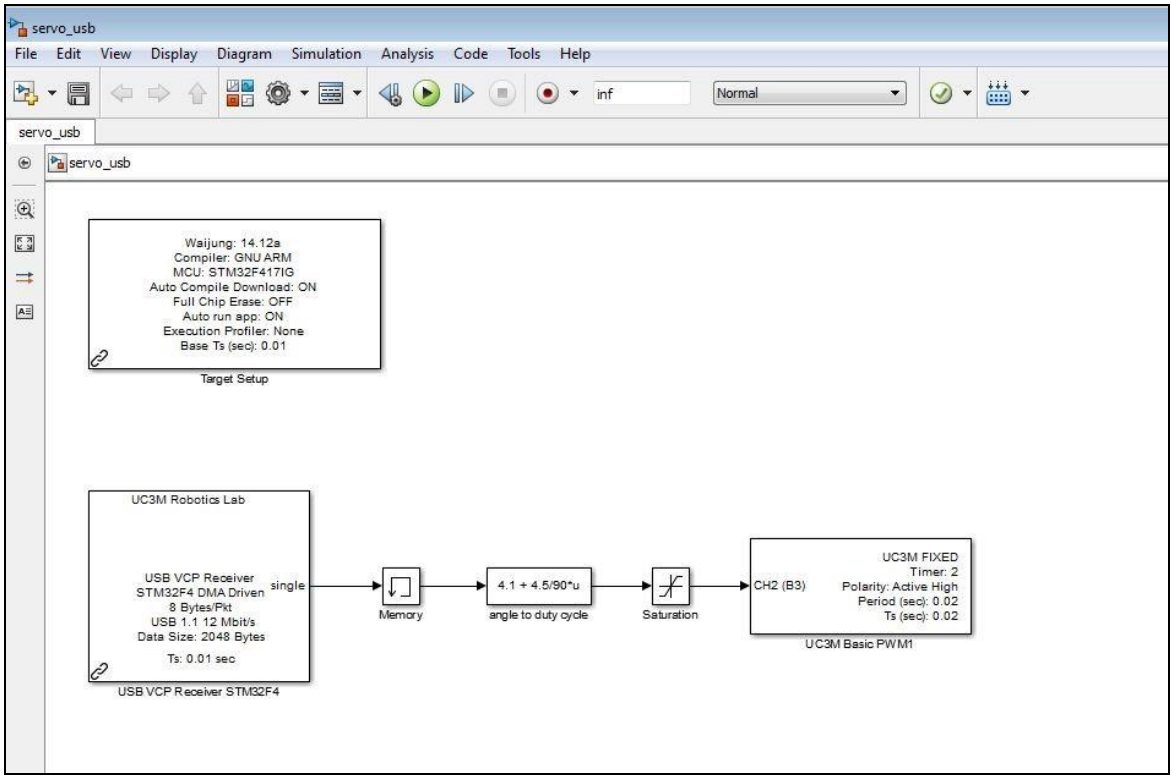


Fig. 4.4: Modelo servo_usb.mdl

A continuación, se explica cada uno de los bloques configurados. Para ello, son necesarias las librerías *Waijung Blockset* y UC3M Addons STM32F4.

En primer lugar, se configuran las diferentes características del microcontrolador con el bloque "*Target Setup"*, tal y como se muestra en la Fig.4.5.

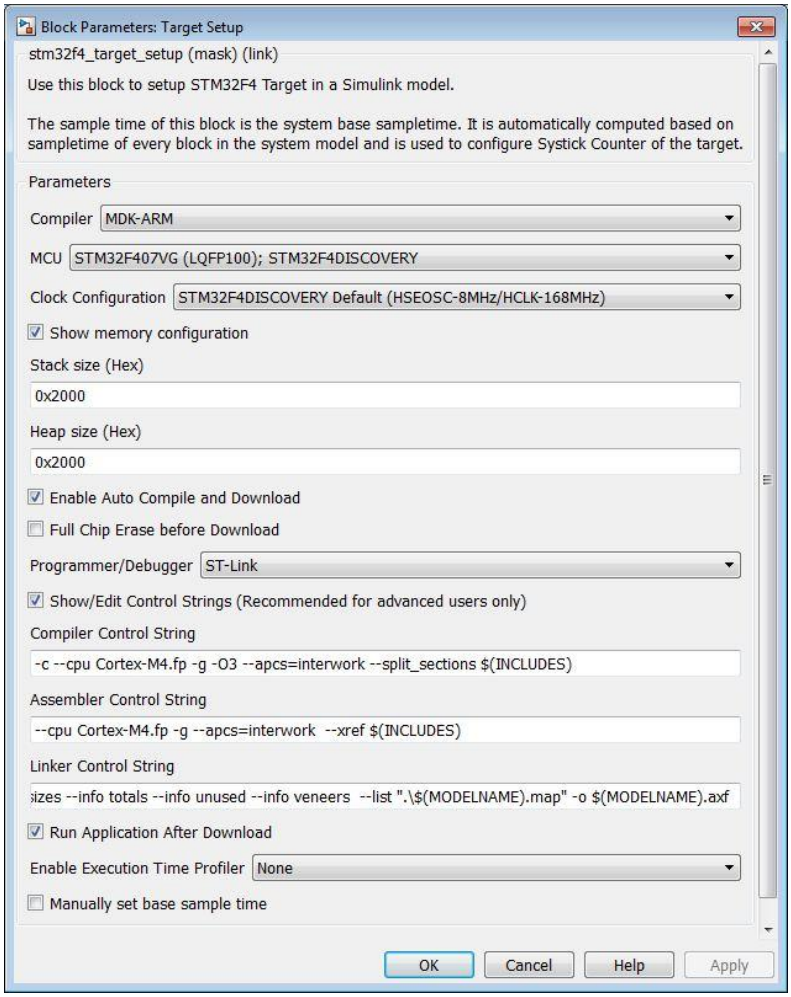


Fig. 4.5: Parámetros del bloque: "Target Setup"

Tras ello, con el bloque *"USB VCP Receiver STMF43F4",* se reciben los datos del ordenador. Para ello, es necesario indicar qué datos se van a utilizar, siendo en este caso una señal de tipo *single*. También, es necesario definir el tiempo de muestreo (0.01) y el inicio y fin de los mensajes. Se puede ver la configuración de este bloque en la Fig. 4.6.

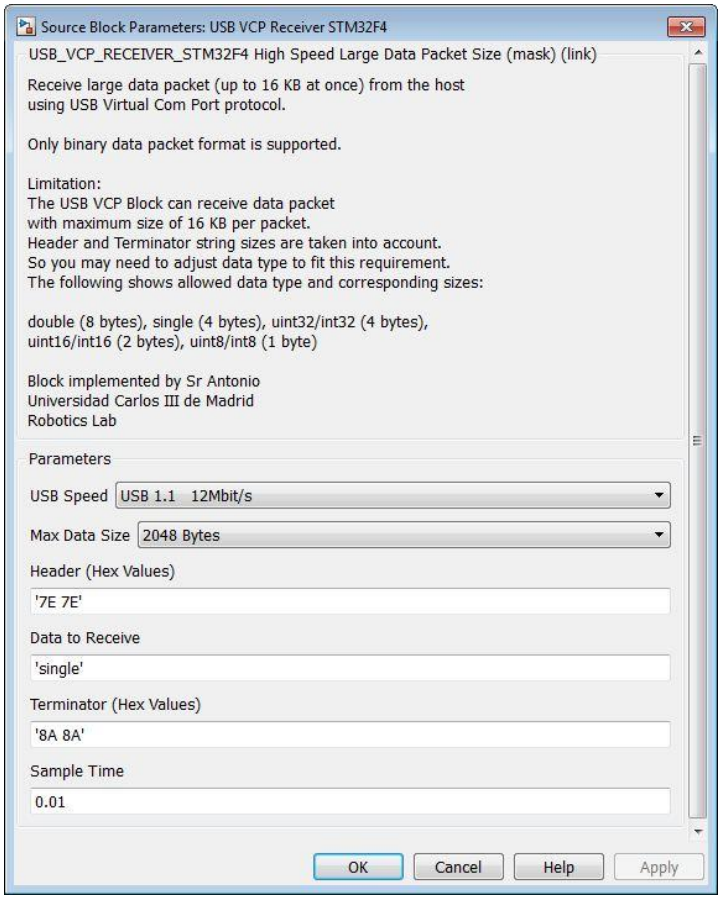


Fig. 4.6: Parámetros del bloque: "USB VCP Receiver STM32F4"

Una vez definido el bloque *"USB VCP Receiver STM32F4"*, se coloca el bloque "Memory" para indicar la posición inicial del servomotor, siendo esta 0.

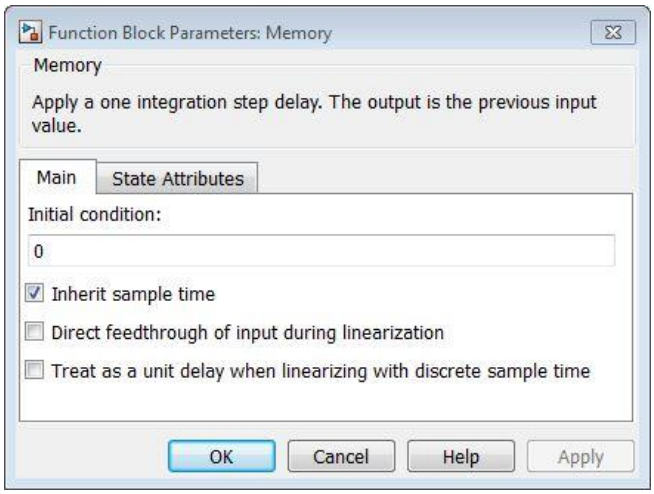


Fig. 4.7: Parámetros del bloque: "Memory"

En el bloque *"angle to duty cicle",* se convierten las señales angulares en valores de ancho de pulso. La variable *u* representa el ángulo final del servomotor. Experimentalmente, para la posición de 0º, se obtiene que el ancho de pulso es 4.1.

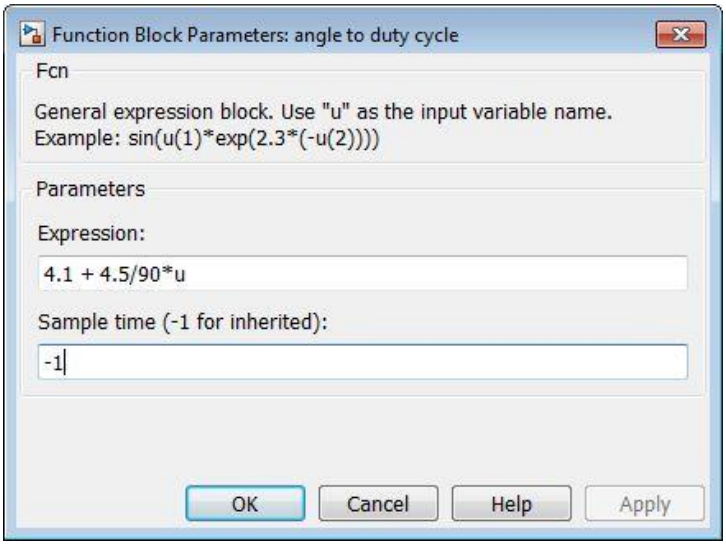


Fig. 4.8: Parámetros del bloque: "angle to duty cycle"

A continuación, se introduce un bloque denominado "*Saturation*", el cual sirve para indicar los límites inferior y superior de la señal y asegurar que el servomotor funciona correctamente.

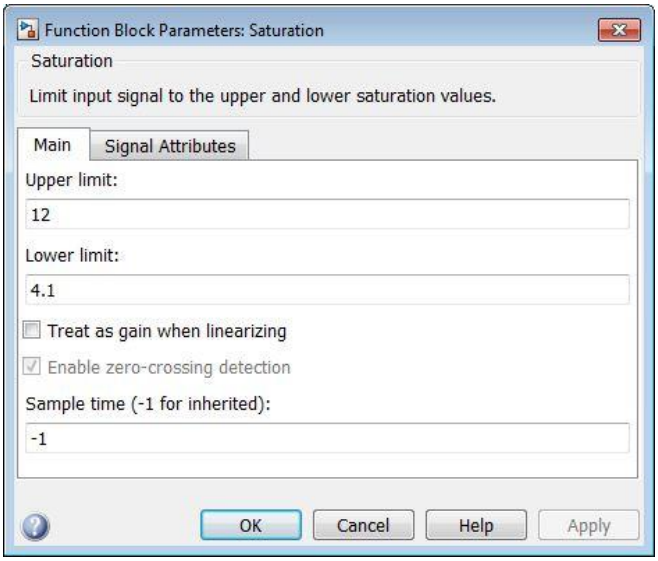


Fig. 4.9: Parámetros del bloque: "Saturation"

Y, para terminar, con el bloque "*UC3M Basic PWM",* se transforman las señales en señales PWM, las cuales son necesarias para controlar el servomotor. En él se indica el período de la señal PWM, el pin utilizado del microcontrolador y el tiempo de muestreo, como se muestra en la Fig.4.10.

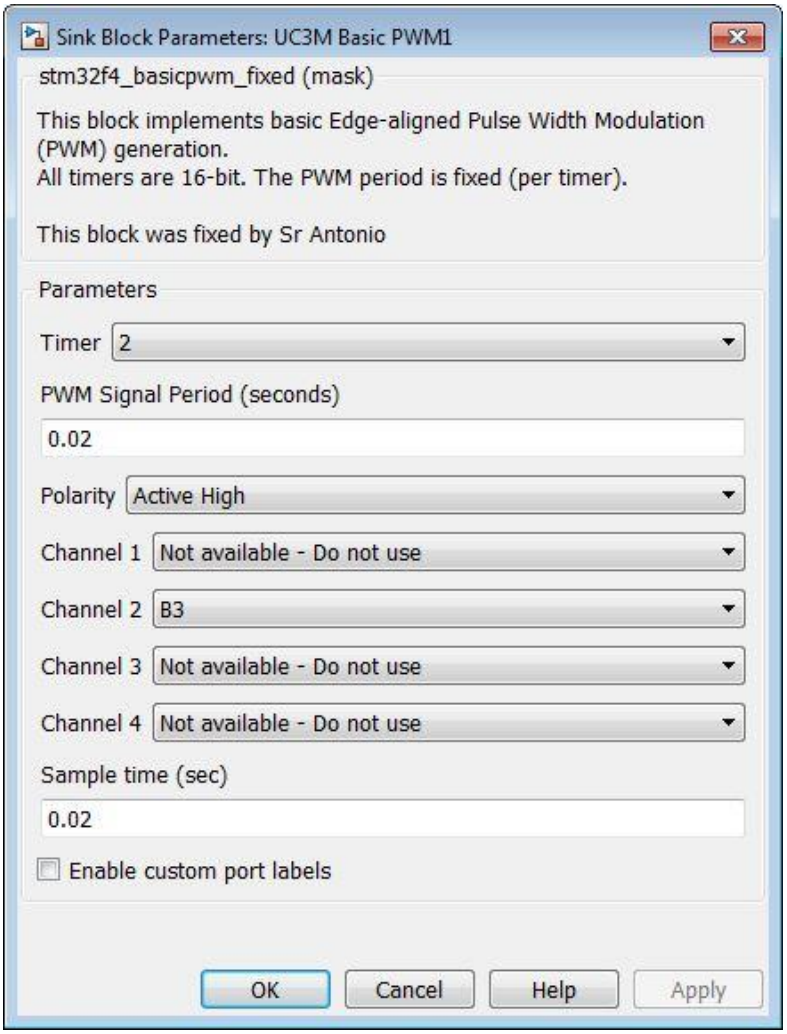


Fig. 4.10: Parámetros del bloque: "UC3M Basic PWM"

4.1.2. Conexión ordenador-servomotor

Una vez programado el controlador, se implementa el modelo *servo_usb_host.mdl*, mostrado en la Fig.4.11, encargado de realizar el control del servomotor.

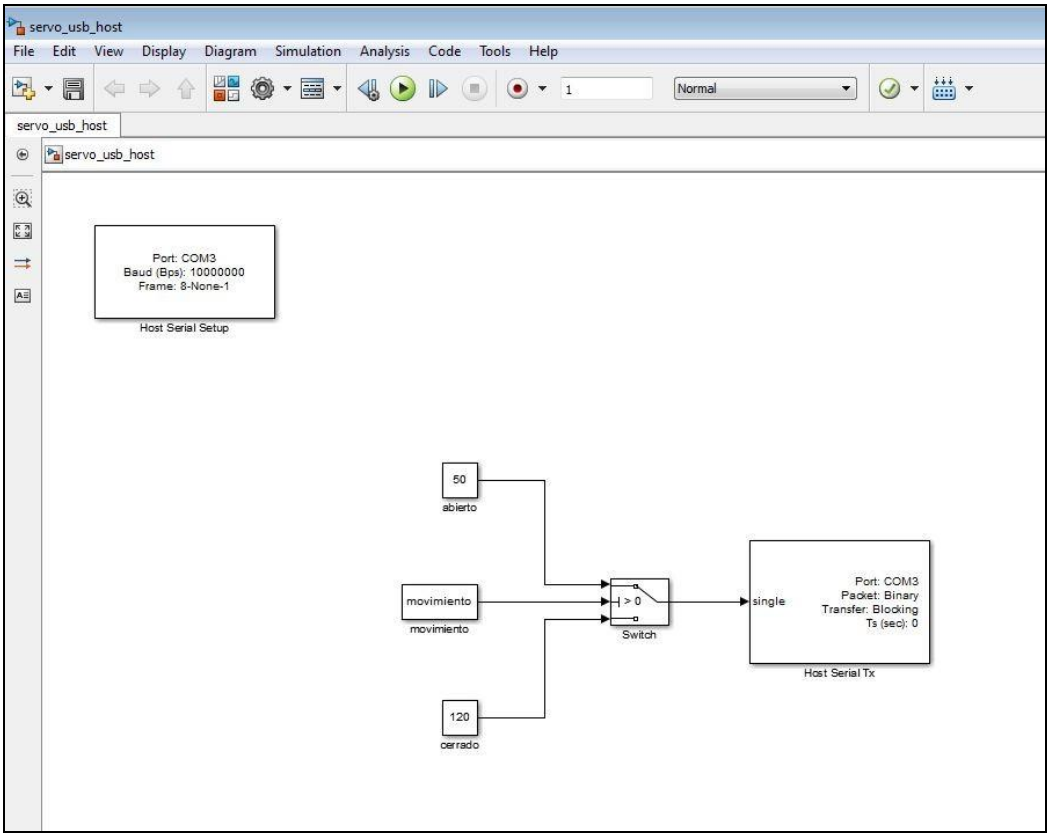


Fig. 4.11: Modelo servo_usb_host.mdl

El primer bloque de este esquema es el bloque "*Host Serial Setup".* En él se selecciona el puerto del ordenador al que está conectado el microcontrolador, como se muestra en la Fig. 4.12.

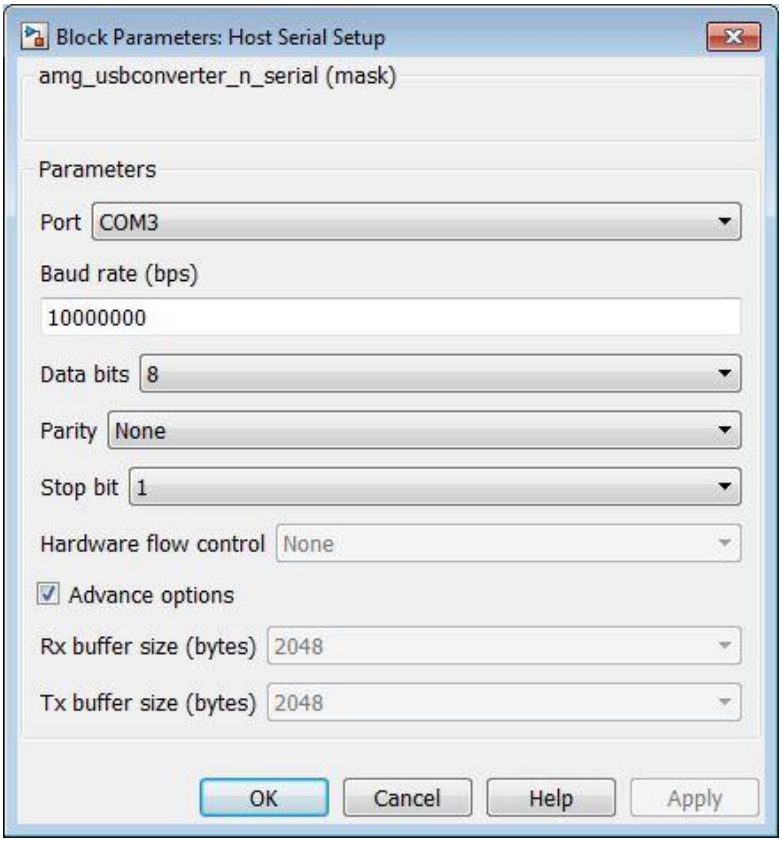


Fig. 4.12: Parámetros del bloque: "Host Serial Setup"

A continuación, se implementa un bloque *switch,* cuya variable de entrada es *movimiento.* Como se ha explicado anteriormente, esta variable vale 1 en caso de que el gesto detectado sea “cerrado” y 0 en el caso de “abierto”. Por tanto, en función de esta variable, se elige el ángulo del servomotor.

El último paso, es incluir el bloque “*Host Serial Tx*”. En él, se selecciona el puerto del ordenador por el que se van a enviar los datos al microcontrolador, así como el tipo de dato que se va a enviar (binario en este caso). También, se indica el inicio y fin de la señal, que será igual al indicado en el modelo *servo_usb* (bloque *“USB VCP Receiver STM32F4”*) y, de igual forma, el número de datos y el tipo que se va a enviar y el tiempo de muestreo. La configuración de estos parámetros se muestra en la Fig.4.13.

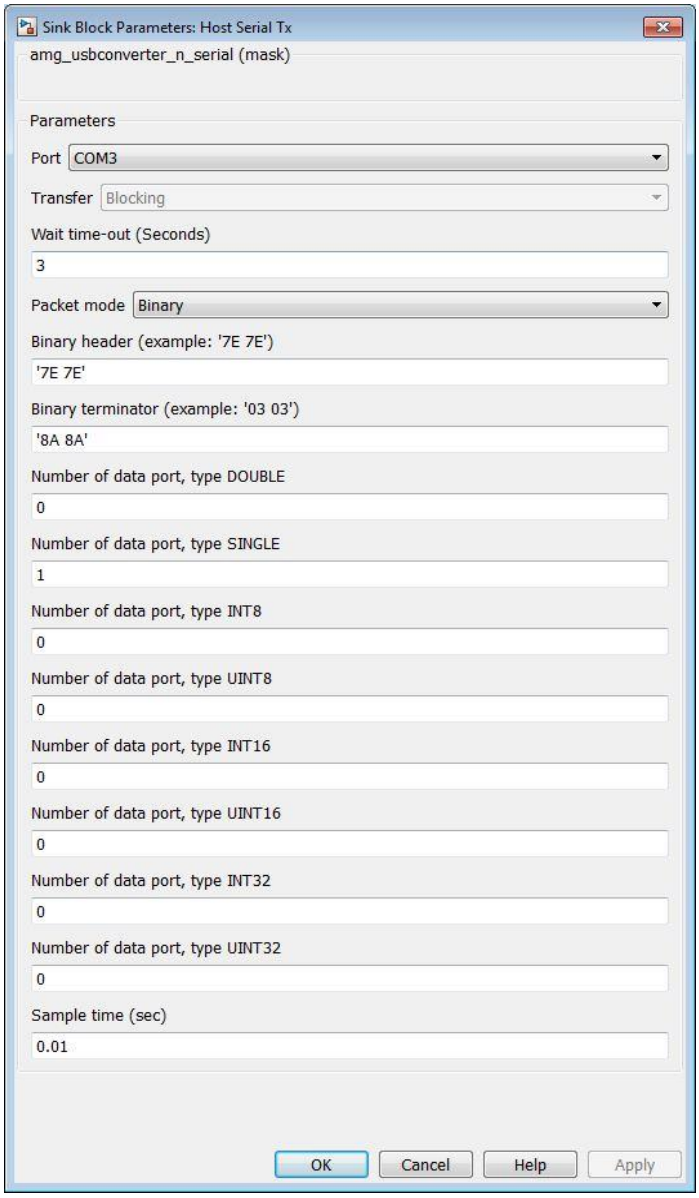


Fig. 4.13: Parámetros del bloque: "Host Serial Tx"

Una vez se programó el microcontrolador y se implementó el modelo Host, se trató de realizar un esquema de Simulink completo. Es decir, incluyendo desde la adquisición de datos hasta la transmisión de la señal al servomotor. Esto no fue posible debido a que no era posible conectar la Myo Armband directamente a Simulink, puesto que su software viene preparado para MATLAB. Por ello, se decidió implementar en código la llamada al modelo Host de Simulink. De manera que, una vez se obtiene el valor de la variable *movimiento*, se simula el modelo *servo_usb_host* desde código en MATLAB, pasándole esta variable y, una vez simulado, continua la ejecución del código *MAIN.m.*

Tras enviar la señal al servomotor, es importante destacar la necesidad de eliminar los datos registrados en esa repetición del bucle utilizando el comando *m1.clearLogs(),* ya que, en caso de no hacerlo, los datos leídos se acumulan y solo reconoce correctamente el primer movimiento.

4.1.3. Optimización del sistema

Como se ha explicado en el apartado anterior, inicialmente se simulaba el modelo de Simulink desde el código de MATLAB para controlar el servomotor. Tras realizar pruebas del sistema completo, se observó que el tiempo de respuesta del servomotor desde que se realizaba el movimiento era superior a los 4 segundos, muy lejos del tiempo real que se pretendía conseguir en este proyecto. Esto era debido a que el programa tenía que abrir y simular el esquema de Simulink.

Por este motivo, se decidió eliminar esta parte del programa y, en lugar de realizar el control del servo mediante Simulink, se implementó en código en MATLAB.

Para ello, se utiliza la función *initializeBC_serial.m.* Esta función recibe como parámetros el puerto COM utilizado y la variable *movimiento*. En primer lugar, define el puerto y la velocidad de comunicación, igual que el bloque "*Host Serial Setup"* que había anteriormente. También, se define el tamaño del Buffer y abre el puerto. A continuación, se escriben los datos en el microcontrolador, indicando el inicio y fin de la señal (de nuevo, igual que en el bloque *"Host Serial Tx"* anterior) y, en función del valor de la variable *movimiento*, se indica un ángulo u otro. En la Fig.4.14 se muestra un flujograma de este código.

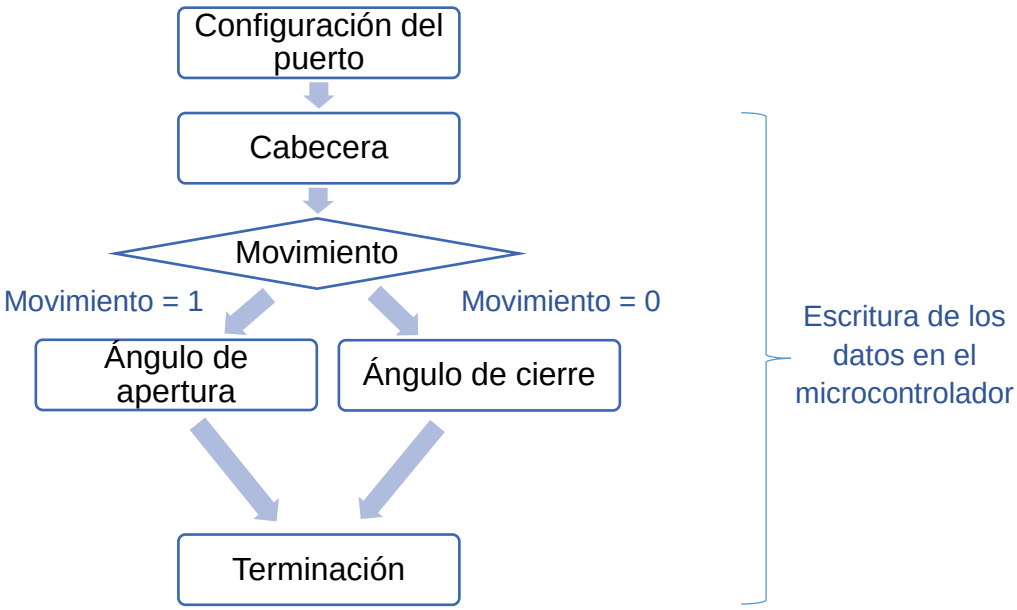


Fig. 4.14: Flujograma del código initializeBC_serial.m

Tras ello, se vuelve a probar la maqueta y, en esta ocasión, se observa que el tiempo de respuesta oscila entre 1 y 2 segundos, por lo que se ha reducido en más de un 50% y, de esta manera, se logra optimizar el sistema.

5. Fabricación de la pinza

Siguiendo el objetivo de realizar una maqueta docente, es necesario fabricar una prótesis con la que los alumnos puedan visualizar el correcto funcionamiento de todo el software desarrollado.

En primer lugar, hay que tener en cuenta que se busca realizar una maqueta de bajo coste, por lo que la impresión 3D es la manera más adecuada de llevarla a cabo. Dado que prevalece el objetivo docente sobre el diseño de una prótesis, se decide tomar un diseño ya realizado pero que se adecúe por completo a las funciones que debe desempeñar.

Por tanto, se ha tomado el diseño realizado por Álvaro Villoslada en su Tesis de Master [26]. Su tesis también se fundamentaba en realizar una prótesis de bajo coste, de manera que el diseño que realizó contaba con una mecánica simple para reducir costes de fabricación y reparación.

El diseño cuenta con tres dedos, simulando los dedo índice, corazón y pulgar en un movimiento de pinza. Estos están diseñados con forma humana, con tres segmentos simulando las falanges (dos en el caso del pulgar), tal y como se muestras en Fig.5.1 y 5.2.

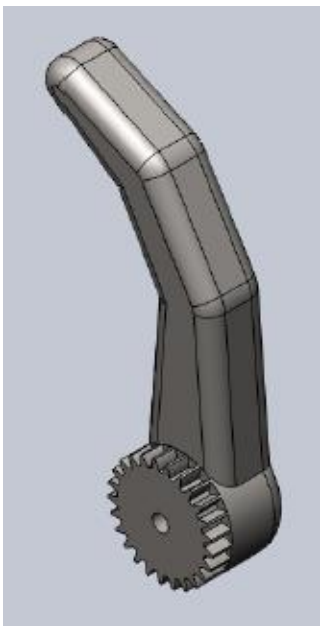


Fig. 5.1: Diseño dedos índice y corazón [26]

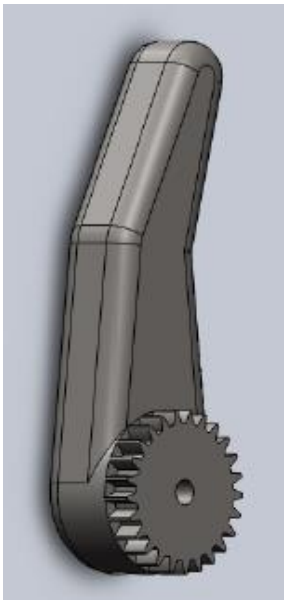


Fig. 5.2: Diseño dedo pulgar [26]

Un sistema de engranajes es el encargado de transmitir el movimiento del servomotor a los dedos. El engranaje del motor mueve otro engranaje prisionero que, por un lado, gira el dedo pulgar y, por otro, gira otro engranaje que está enganchado al engranaje de los dedos índice y corazón. Gracias a ello, se puede realizar el movimiento de apertura y cierre de la pinza con los ángulos deseados. En la Fig. 5.3, se presenta el conjunto de engranajes.

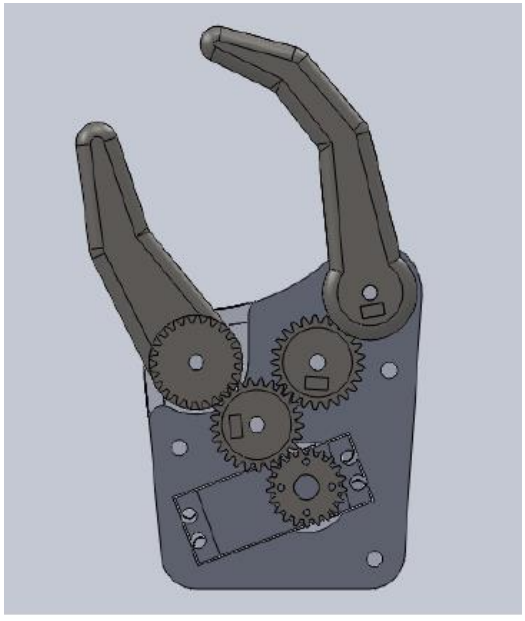


Fig. 5.3: Diseño del conjunto de engranajes [26]

Todo el conjunto está rodeado por dos carcasas, mostradas en la Fig. 5.4, encargadas de cubrir el servomotor. En ellas, se encuentran los rodamientos de los ejes de los dedos, dotándoles de mayor agilidad en el movimiento.

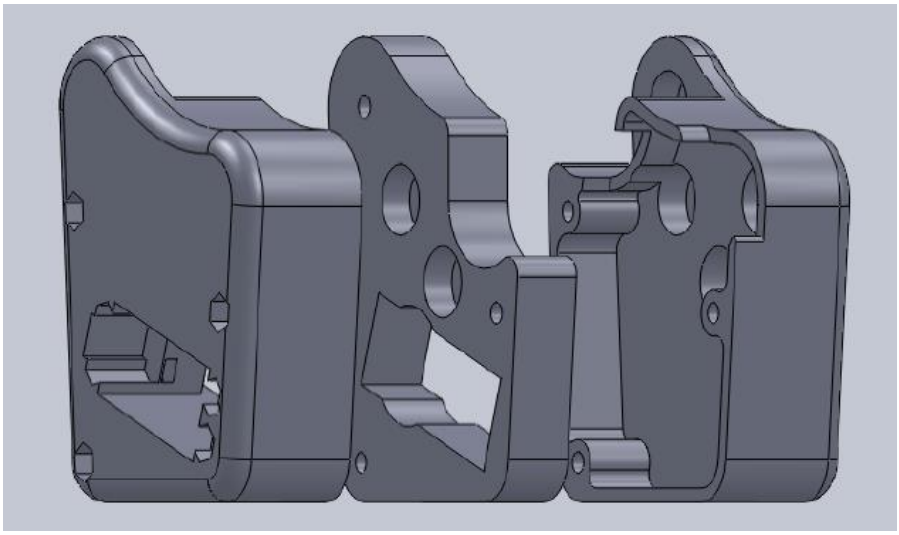


Fig. 5.4: Diseño carcasas [26]

En la Fig. 5.5, se muestra el diseño completo de la pinza.

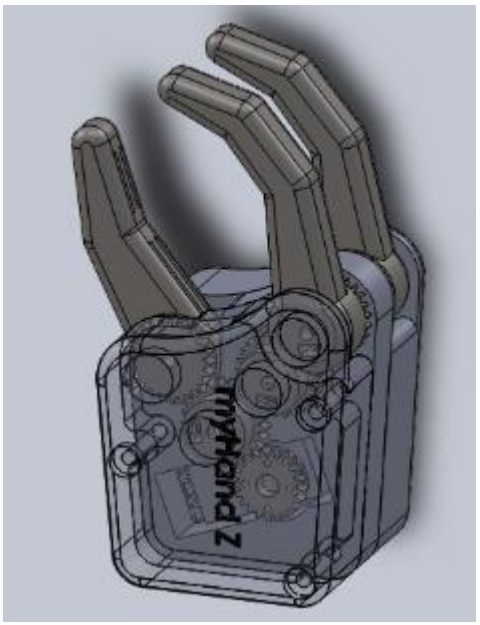


Fig. 5.5: Diseño completo de la pinza [26]

Como se ha mencionado anteriormente, la fabricación de la pinza se realiza mediante impresión 3D con plástico ABS por ser el medio más económico.

En las Fig. 5.6, 5.7 y 5.8 se puede ver el montaje de las piezas y, en la Fig. 5.9, el resultado final.

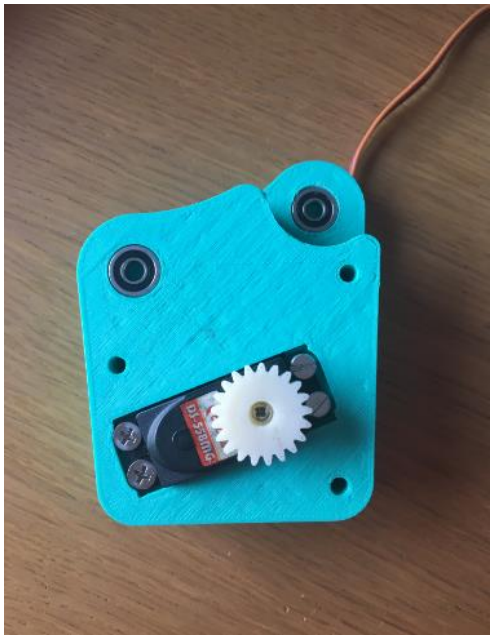


Fig. 5.6: Base principal con servomotor

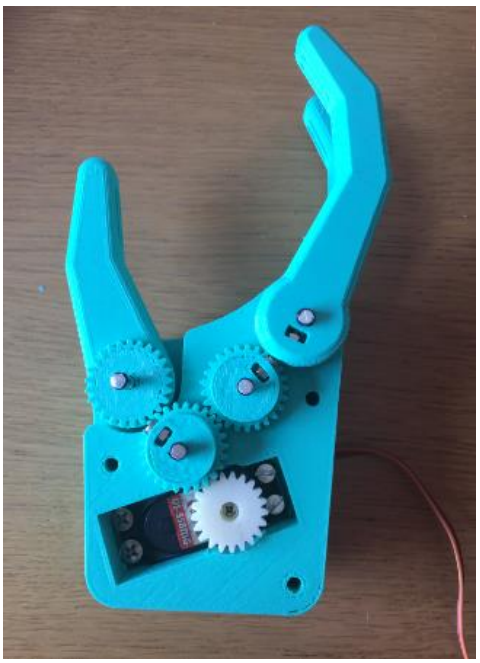


Fig. 5.7: Sistema de engranajes con la pinza abierta

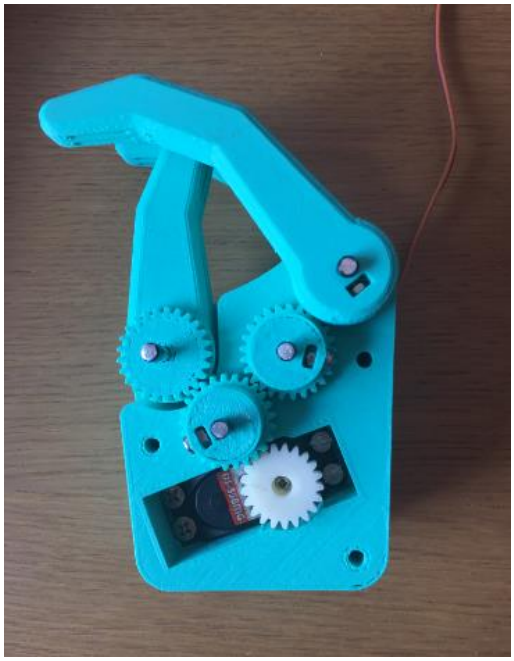


Fig. 5.8: Sistema de engranajes con la pinza cerrada

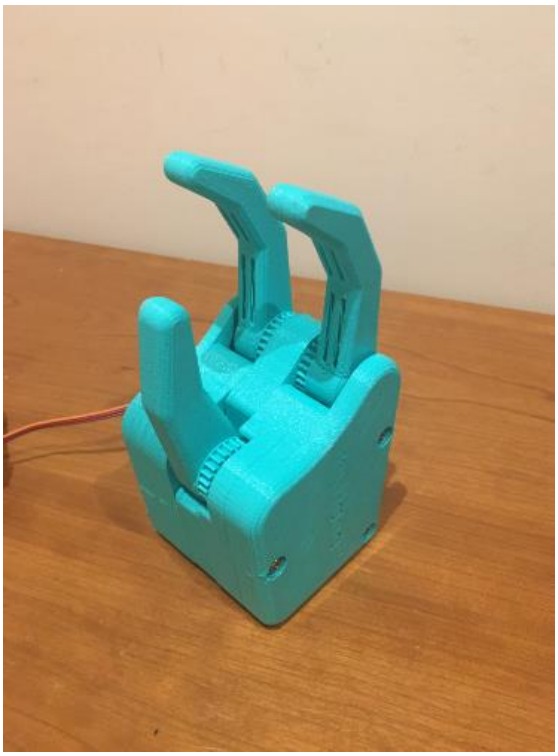


Fig. 5.9. Pinza completa montada

6. Guion de prácticas

En este capítulo, siguiendo el objetivo de que la maqueta sirva como elemento en prácticas de laboratorio, se presenta un guion de prácticas para la asignatura de "Aplicaciones de la automática en biomédica". Dado que en una primera práctica de la asignatura se realiza el control de un servomotor aplicando un umbral, se aprovecha esta práctica para que el alumno conozca la técnica de reconocimiento de patrones y, así, pueda aprender desde cómo se realiza la adquisición de señales EMG hasta la clasificación del movimiento mediante esta técnica.

6.1. Práctica: Reconocimiento de patrones en señales EMG

6.1.1. Objetivo de la práctica

El objetivo de esta práctica es controlar el movimiento de una pinza mediante la clasificación de los movimientos de apertura y cierre de la mano utilizando la técnica de reconocimiento de patrones.

Esta técnica se base en extraer determinadas características de las señales de electromiografía adquiridas y, con ellas, clasificar el gesto realizado. En la Fig. 6.1 se muestran los pasos a realizar para llevar a cabo esta técnica.

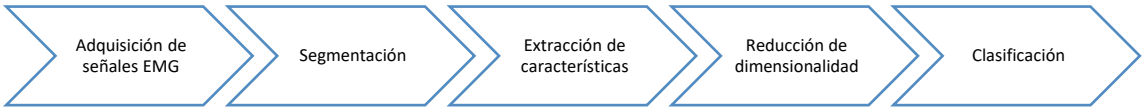


Fig. 6.1: Etapas del reconocimiento de patrones

6.1.2. Materiales

Para llevarla a cabo, se cuenta con los siguientes materiales:

- Ordenador con MATLAB
- Myo Armband
- Microcontrolador STMF32F4
- Pinza con servomotor
- Cables de conexión del microcontrolador

6.1.3. Configuración del Myo Armband

En primer lugar, antes de comenzar con el desarrollo de la práctica es necesario sincronizar y calibrar el Myo Armband.

Para ello, se abre el programa Myo Connect.

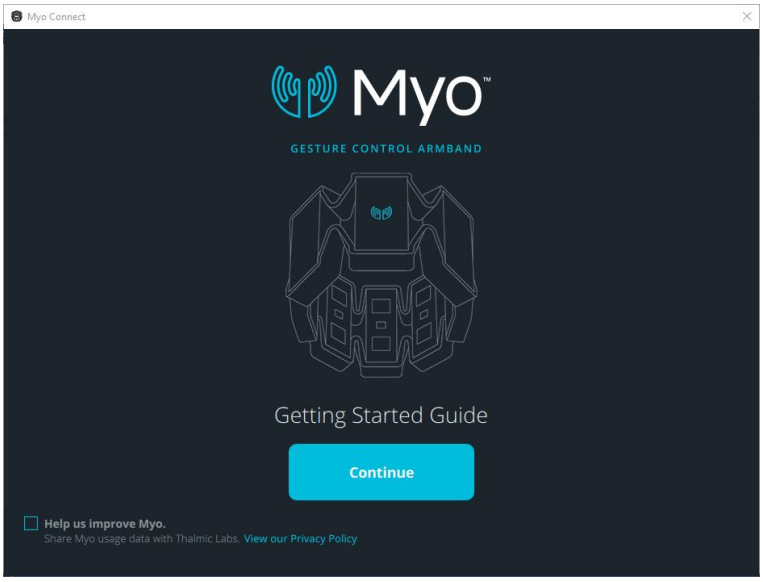


Fig. 6.2: Myo Connect

Una vez abierto, se conecta el cable USB al MYO y el adaptador Bluetooth como se indica.

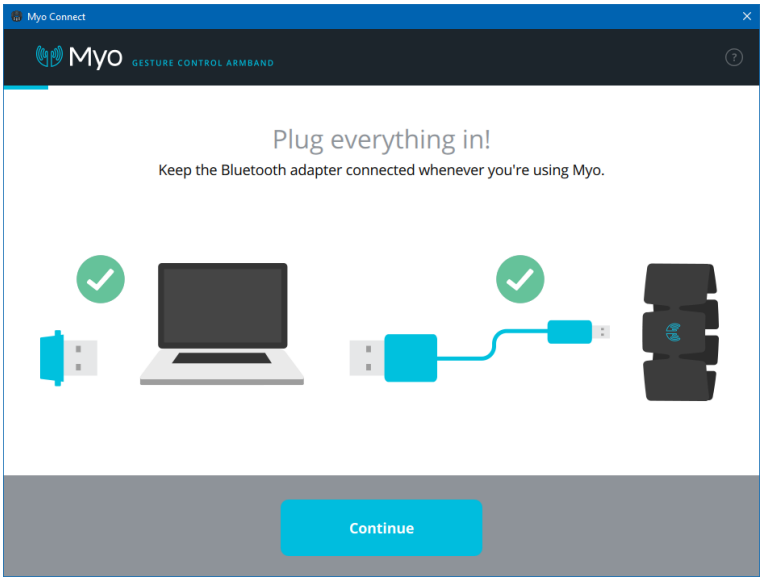


Fig. 6.3: Conexión del Myo Armband

El siguiente paso es poner el nombre del dispositivo. Cada grupo lo nombrará de la siguiente manera: *Grupo_X* (siendo X el número de grupo asignado al inicio de la práctica).

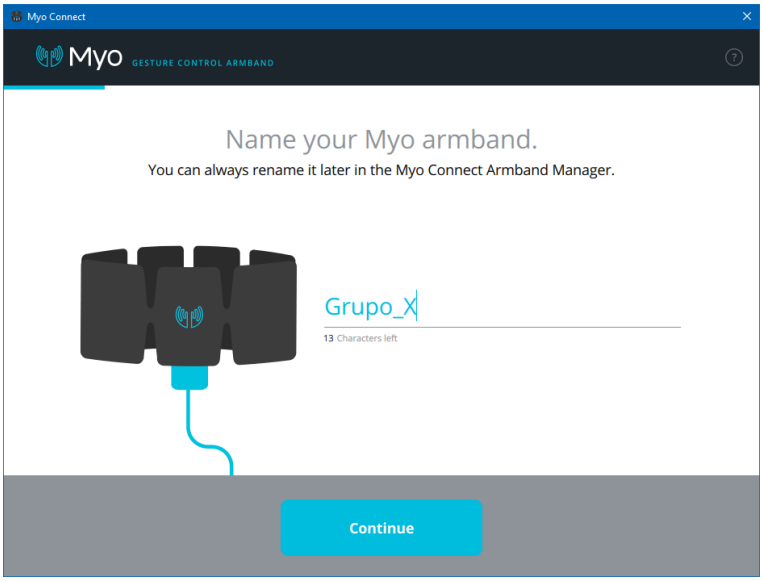


Fig. 6.4: Nombre del dispositivo Myo Armband

A continuación, se desconecta el cable USB del MYO Armband.

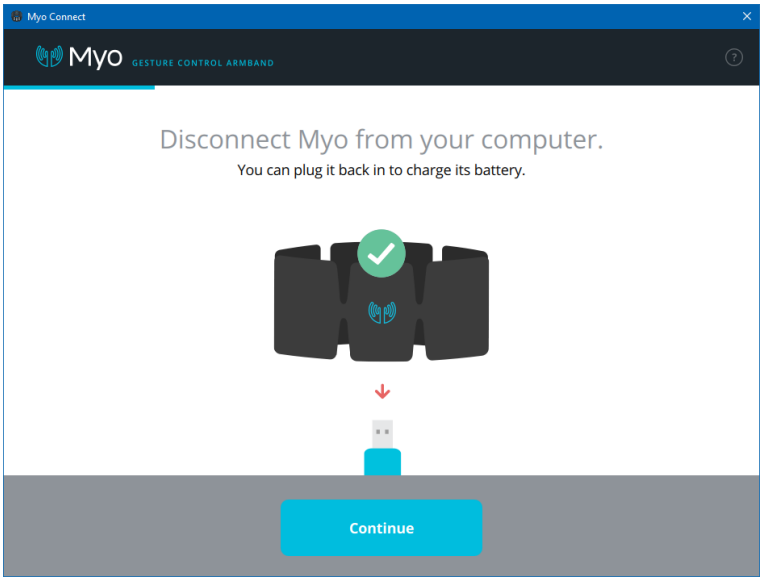


Fig. 6.5: Desconexión del cable USB del Myo Armband

Antes de sincronizar el Myo, hay que colocarlo en el antebrazo con el electrodo del LED en la parte superior, como se muestra a continuación.

Fig. 6.6: Colocación del Myo Armband en el antebrazo

Una vez colocado correctamente, es hora de sincronizarlo. Para ello, se mueve la mano hacia fuera, como se muestra en la imagen.

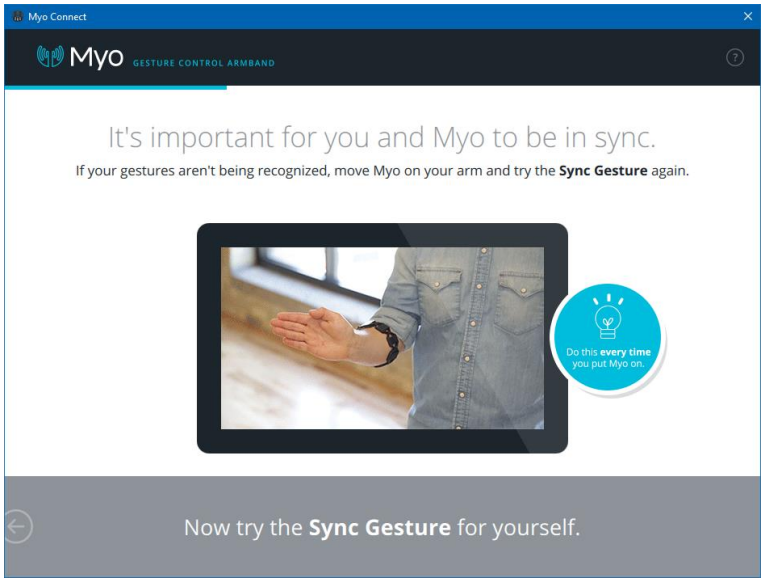


Fig. 6.7: Sincronización del Myo Armband

Una vez sincronizada, se esperan unos segundos hasta que el dispositivo esté preparado y, para terminar, se realizan los movimientos que se indican en la pantalla.

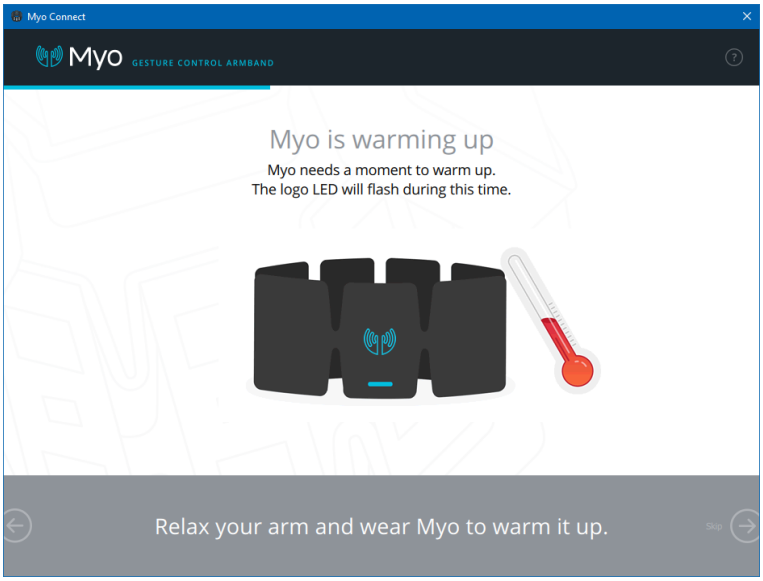


Fig. 6.8: Pantalla de espera tras sincronización del Myo Armband

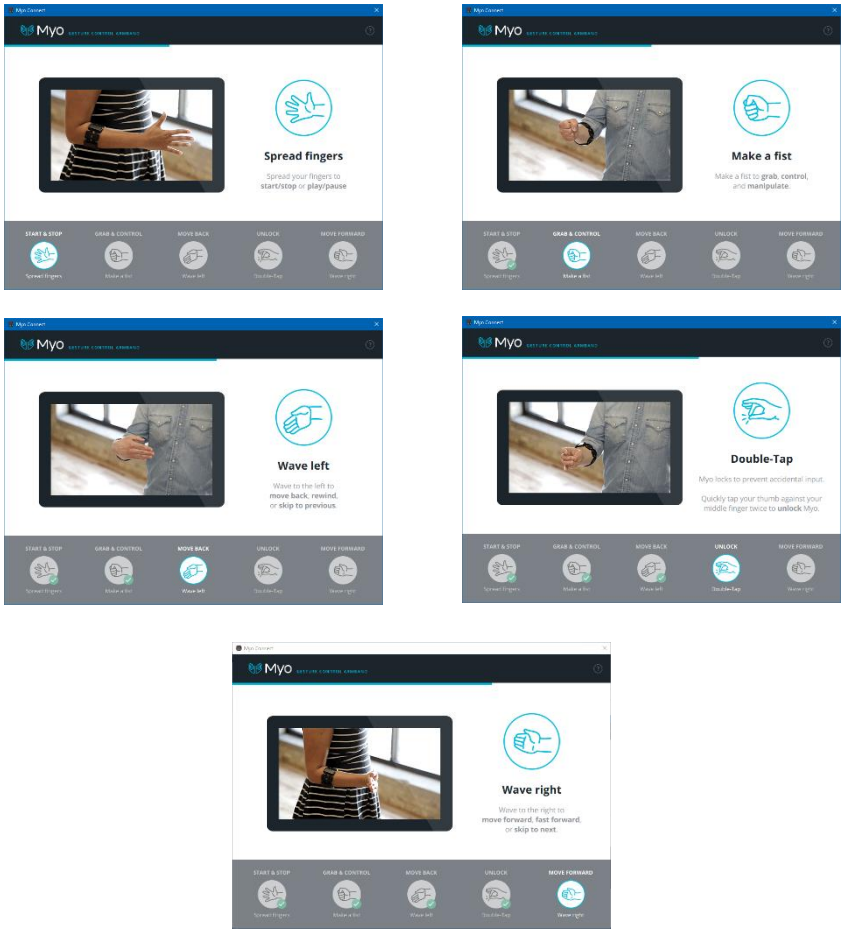


Fig. 6.9: Calibración del Myo Armband

6.1.4. Adquisición de señales

Una vez se ha sincronizado el Myo Armband, se van a adquirir varias señales con las que, posteriormente, se entrenará el clasificador.

Para ello, se descarga de Aula Global el fichero *Quickstart_EMG_desagregados.m.*

En esta ocasión se utilizarán 5 medidas para cada movimiento. Con el comando pause(x), se indica el tiempo de muestreo. Para esta primera fase de adquisición de datos se tomarán señales durante 5 segundos.

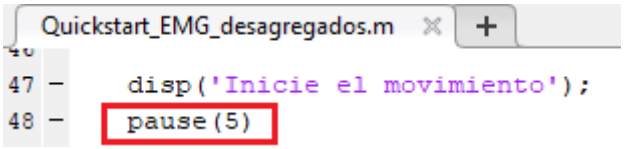


Fig. 6.10: Muestra de código para indicar el tiempo de muestreo

Es necesario ir guardando cada señal. Para ello, se creará una carpeta llamada "DATOS" y, en cada adquisición, se modifica la siguiente línea:

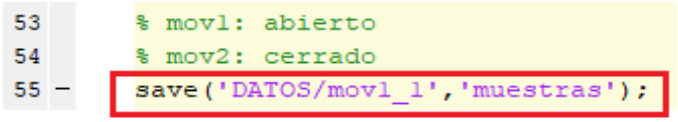


Fig. 6.11: Muestra de código para guardar los datos adquiridos

En ella se puede observar que el movimiento 1 será la mano abierta y, el 2, con el puño cerrado. Por tanto, el nombre del archivo será *movX_Y*, donde X es el número del movimiento e Y la iteración correspondiente.

Para realizar la adquisición, se pulsa el botón Run en la parte superior de la pantalla.

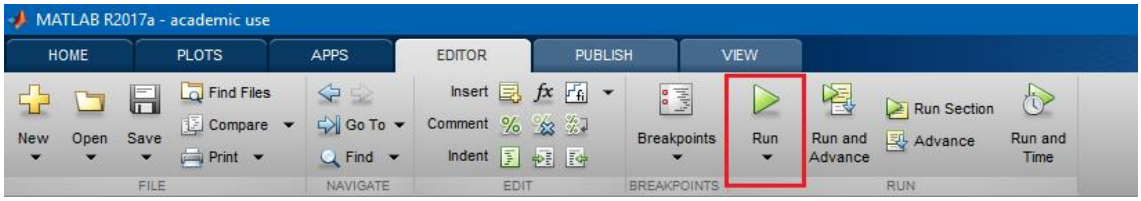


Fig. 6.12: Botón Run

Al final de cada adquisición, se muestra la figura correspondiente a la señal leída por cada uno de los 8 electrodos. De esta manera, se pueden ver los músculos que se activan con cada tipo de movimiento.

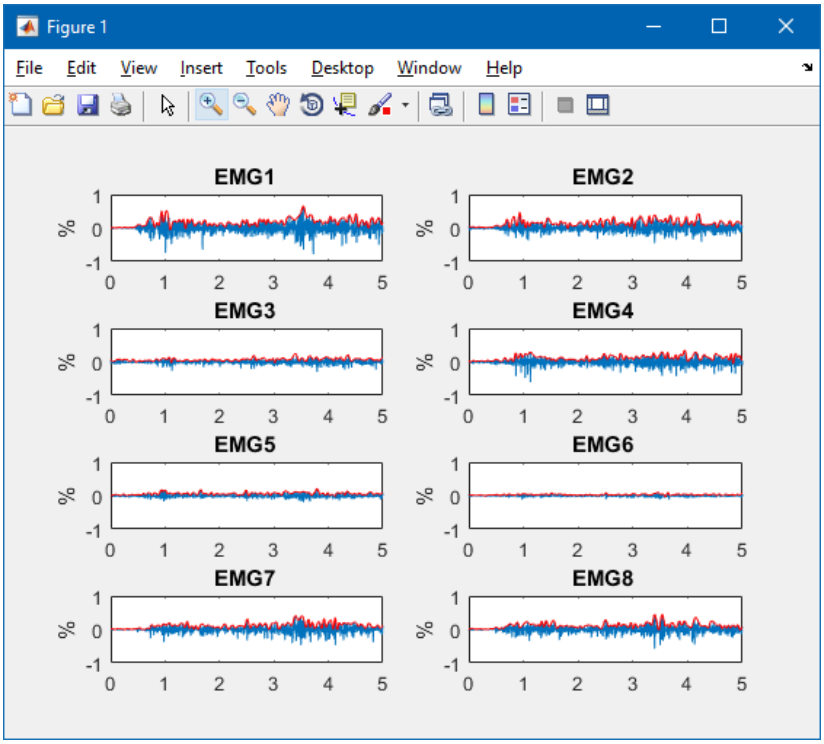


Fig. 6.13: Ejemplo de gráfica de señal EMG separada en 8 canales

Ejercicio 1: *Tomar datos con diferentes movimientos, además de apertura y cierre, y realizar una comparación con las diferentes figuras obtenidas. En función del movimiento realizado, analizar qué canal es el más indicado para identificarlo.*

6.1.5. Procesamiento de las señales

Una vez adquiridos todos los datos, es necesario procesar las señales. En primer lugar, es necesario separar la señal en los 8 canales y recortarlas tanto al principio como al final para quedarnos con 8 vectores de 900 muestras cada uno. Esto se puede realizar de la siguiente manera:

```
mov1_ch1_it1=muestras(:,1);
mov1_ch1_it1_recortada=mov1_ch1_it1(91:990);
```

Fig. 6.14: Ejemplo de código para separar y recortar las señales

Como se observa, con la primera línea se guarda en la variable *mov1_ch1_it1* las muestras del primer canal para la primera iteración y, a continuación, se eliminan las 90 primeras muestras de la señal y las posteriores a las 990. Esto es necesario para todos

los movimientos, canales e iteraciones. Obteniendo como resultado matrices de 900x5 (900 muestras y 5 iteraciones).

A continuación, todos los datos correspondientes a un movimiento y canal, se unen en una sola matriz, de la siguiente manera:

```
mov1_ch1=[mov1_ch1_it1_recortada mov1_ch1_it2_recortada ... mov1_ch1_it5_recortada];
```

Fig. 6.15: Ejemplo de código para matrices tipo movX_chY

Por tanto, se tienen 16 matrices (2 movimientos x 8 canales) de dimensiones 900x5 (900 muestras y 5 iteraciones). Estas matrices se llaman *movX_chY*, donde X es el tipo de movimiento e Y el canal.

Una vez se tienen estas matrices, se descarga el archivo *time_domain_feature_extraction.m* de Aula Global. Este código será el encargado de segmentar la señal, extraer características y hacer la reducción de dimensionalidad PCA.

Tras ejecutarlo, se habrán guardado las matrices *feat_pca* y *target*. Con las que se va a entrenar la red neuronal.

Además, se muestra una figura en la que se puede observar la reducción a dos dimensiones que se ha llevado a cabo. En esta gráfica se deben ver claramente separados los datos en dos grupos. De esta manera se puede predecir si el clasificador podrá diferenciar claramente los dos movimientos.

Ejercicio 2: *Procesar las señales y analizar la gráfica obtenida tras la reducción de dimensionalidad*

6.1.6. Clasificación

Una vez procesada la señal, ya se puede llevar a cabo la clasificación. En esta ocasión, se va a generar y entrenar una red neuronal de la manera que se explica a continuación:

```
l1=load('FEAT_PCA');
l2=load('TARGET');

feat_pca=l1.feat_pca;
target=l2.target;
net = feedforwardnet;
net = train(net,feat_pca,target);

save('net');
```

Fig. 6.16: Código de generación y entrenamiento de la red neuronal

En primer lugar, se cargan las matrices *feat_pca* y *target* obtenidas en el apartado anterior. Con el comando *feedforwardnet* se genera la red y se entrena en la siguiente línea como se muestra en la imagen.

Una vez se ha entrenado la red, ya se puede identificar el movimiento que se ha realizado según la salida de la misma.

Para este último paso, se utiliza el código *MAIN.m* de Aula Global. Con él va a ser posible adquirir señales a tiempo real y, también, procesarlas y clasificar el movimiento.

```
l=load('NET');
net=l.net;
resultado = sim ( net,feat_pca);

if resultado(1) > resultado (2)
    disp('abierto');
    movimiento = 0;

else
    disp ('cerrado');
    movimiento = 1;
end
```

Fig. 6.17: Fragmento de código MAIN.m para realizar la clasificación

La variable *movimiento* vale 0 si el movimiento es de apertura y 1 si es de cierre.

Ejercicio 3: *Ejecutar el código MAIN.m y comprobar que la clasificación de movimiento se realiza correctamente*

6.1.7. Prueba en sistema físico real

Una vez sabemos que la clasificación se realiza correctamente, se procede a conectar la maqueta.

El microcontrolador STM32F4 ya está programado con una salida PWM en el pin B3. A continuación, se indican los pasos para conectar la maqueta:

1. Conectar el cable negro a GND en el microcontrolador
2. Conectar el cable amarillo a 5V en el microcontrolador
3. Conectar el cable blanco al pin B3 del microcontrolador
4. Conectar el cable negro del microcontrolador al marrón del servomotor
5. Conectar el cable amarillo del microcontrolador al rojo del servomotor
6. Conectar el cable blanco del microcontrolador al naranja del servomotor
7. Conectar el cable micro USB del ordenador al microcontrolador

8. Conectar el cable mini USB del ordenador al microcontrolador

Una vez conectados todos los elementos, se descarga de Aula Global el archivo *initializeBC_serial.m*.

Dentro del código *MAIN.m* utilizado anteriormente, se descomenta la línea en la que se realiza la llamada a *initializeBC_serial.m.*

```
initializeBC_serial(6,movimiento);
```

Fig. 6.18: Llamada a la función initializeBC_serial

El primer parámetro es el número del puerto COM del ordenador al que está conectado el microcontrolador. Para ver qué puerto se está utilizando, dentro del Panel de Control se selecciona "*Ver dispositivos e impresoras*".

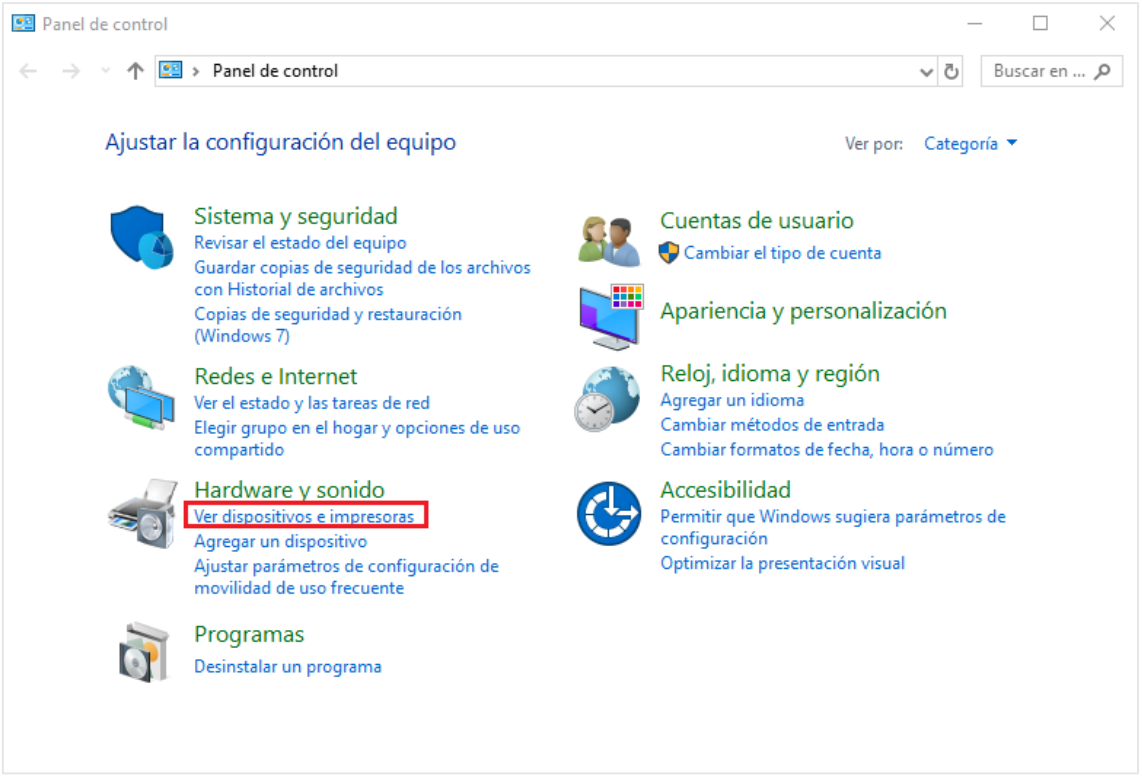


Fig. 6.19: Panel de control

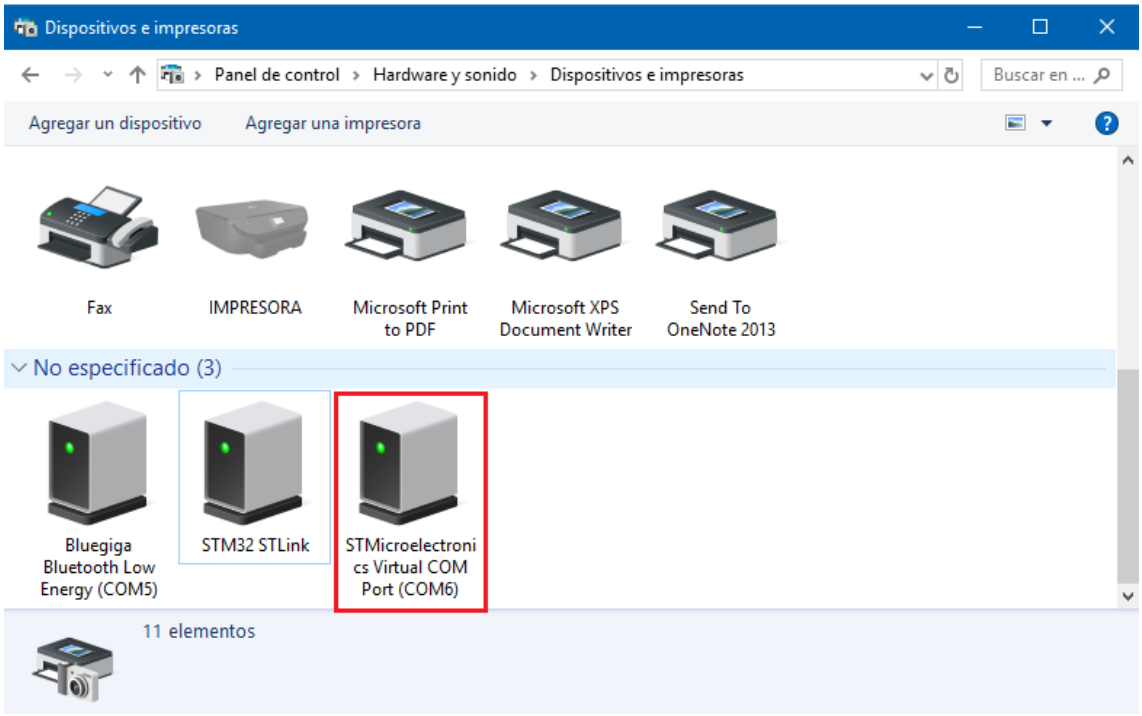


Fig. 6.20: Puerto COM

Una vez indicado el puerto COM, en el código *initializeBC_serial.m*, se pueden modificar los ángulos que se desean para cada uno de los movimientos.

```
if movimiento==0
 fwrite(serPort,120,'single');
elseif movimiento==1
  fwrite(serPort,40,'single');
end
```

Fig. 6.21: Muestra código initializeBC_serial.m para seleccionar ángulo

Ejercicio 4: *Integrar todos los elementos y realizar la adquisición a tiempo real de ambos movimientos modificando los valores de los ángulos para ver el funcionamiento del sistema completo. Por último, con la red entrenada por uno de los alumnos, comprobar el funcionamiento de la misma con el otro miembro del grupo e interpretar los resultados.*

7. Resultados

En este capítulo se presentan los resultados obtenidos tras la realización de diferentes pruebas para evaluar el correcto funcionamiento la red neuronal o el funcionamiento a tiempo real de la maqueta. También, se presenta la maqueta completa.

7.1. Evaluación del funcionamiento de la red neuronal

Una vez entrenada la red, MATLAB permite extraer diferentes gráficos. En la Fig. 7.1, se muestra el histograma de error. En él se observa que la mayor parte de datos cae muy cerca de la línea de error cero, lo que significa que no existirá prácticamente error en la clasificación. Además, en la Fig. 7.2, se pueden ver las gráficas de regresión obtenidas. Con ellas se pretende ver que la salida de la red se ajusta al objetivo, por lo que, si la recta de puntos presenta una gran similitud con la recta de color, significa que la red realizará la clasificación correctamente. Además, se observa un valor de R del 99,115%, por lo que se puede concluir que la clasificación se hará de forma idónea.

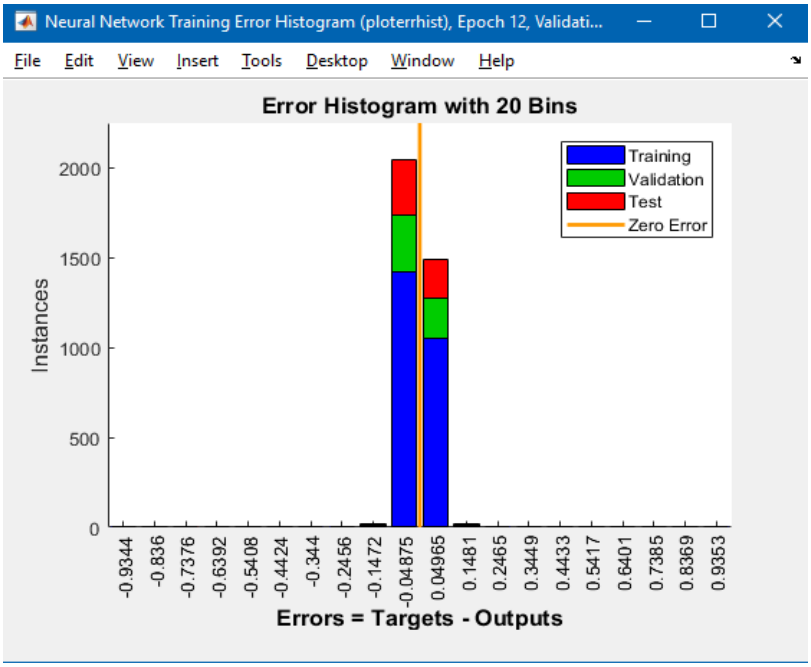


Fig. 7.1: Histograma de error

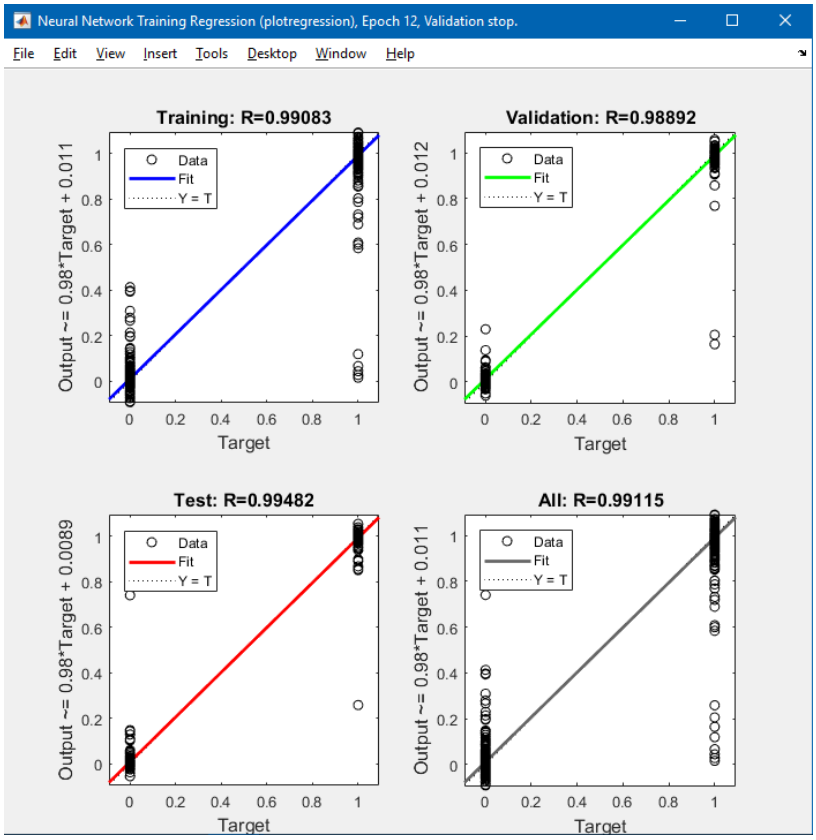


Fig. 7.2: Regresión

A continuación, se prueba la red neuronal con nuevos datos de manera que se adquieren señales de dos usuarios diferentes. Esto se realiza en cinco series de diez iteraciones cada una en dos días diferentes. La primera y la segunda serie constaban de un único movimiento, la tercera y la cuarta se realizaron con ambos movimientos intercalados y, la última, con un orden aleatorio de los mismos. En las Fig. 7.3 y 7.4, se pueden observar los resultados obtenidos.

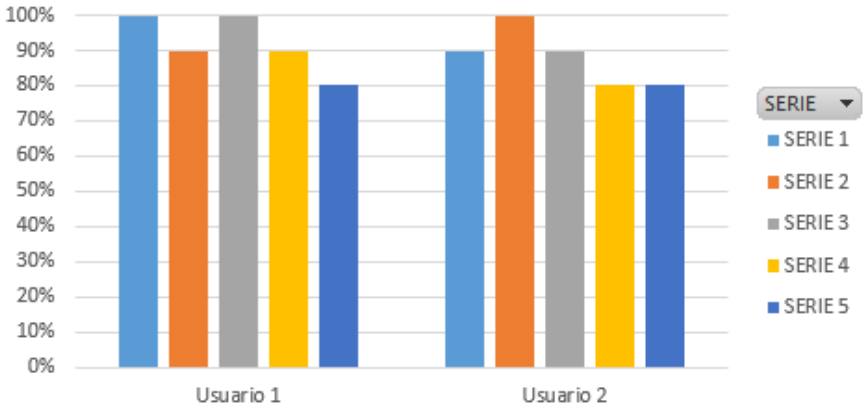


Fig. 7.3: Gráfica de evaluación de la red neuronal en la sesión 1

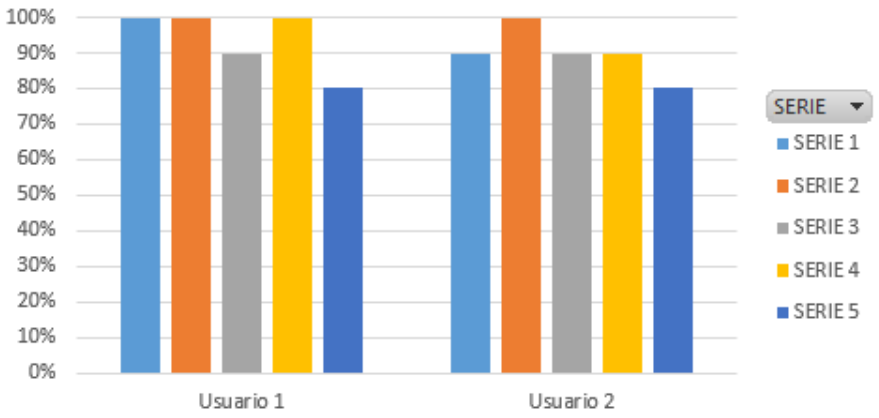


Fig. 7.4: Gráfica de evaluación de la red neuronal en la sesión 2

En las gráficas anteriores, se puede observar que el error aumenta a medida que se realizan más series. Esto se debe, en gran medida, a la fatiga muscular, la cual afecta considerablemente a las señales de electromiografía.

También, se observa que la tasa de error en el usuario 2 es ligeramente superior (12% frente a un 8% en la primera sesión y 10% frente a un 6% en la segunda), debido a que las señales usadas para entrenar la red neuronal fueron las del usuario 1 por lo que, aunque reconoce correctamente un alto porcentaje de movimientos en el usuario 2, no es igual de efectiva. Otro posible motivo es, que el usuario 2 no se haya colocado de manera correcta el Myo Armband, al ser su primer contacto con él pero, dado que la diferencia es únicamente de un 4%, se considera que la red clasifica correctamente independientemente del usuario.

Otro dato a tener en cuenta es que los errores producidos son siempre identificando el gesto de abierto como si fuera cerrado. Esto ocurre sobre todo durante la ejecución a tiempo real, ya que se toman muestras únicamente durante 0.5 segundos y la evaluación para llevar a cabo la clasificación se realiza en función del primer valor de cada fila de la matriz resultado obtenida de la red neuronal. Por lo que, si la lectura de datos comienza durante la transición del movimiento, es probable que se registre cierta actividad muscular al inicio del movimiento y que este se detecte como cerrado. Por ello, a veces es necesaria una segunda iteración para que la pinza abra correctamente cuando le corresponde.

7.2. Evaluación del funcionamiento a tiempo real

Uno de los objetivos del proyecto era el funcionamiento de la maqueta a tiempo real. En el capítulo 4 se explica cómo se ha optimizado el sistema, reemplazando el uso de Simulink por código de MATLAB, logrando disminuir un tiempo promedio de ejecución de 4.3 a 1.2 segundos, tal y como se explicará a continuación.

Para medir el tiempo de ejecución se utilizan las funciones *tic toc* de MATLAB, las cuales permiten medir el tiempo de fragmentos de código.

Para ello, se han realizado 10 series de 10 iteraciones con cada una de las partes del código: adquisición de datos, procesamiento de señales, clasificación y envío de la señal al servomotor.

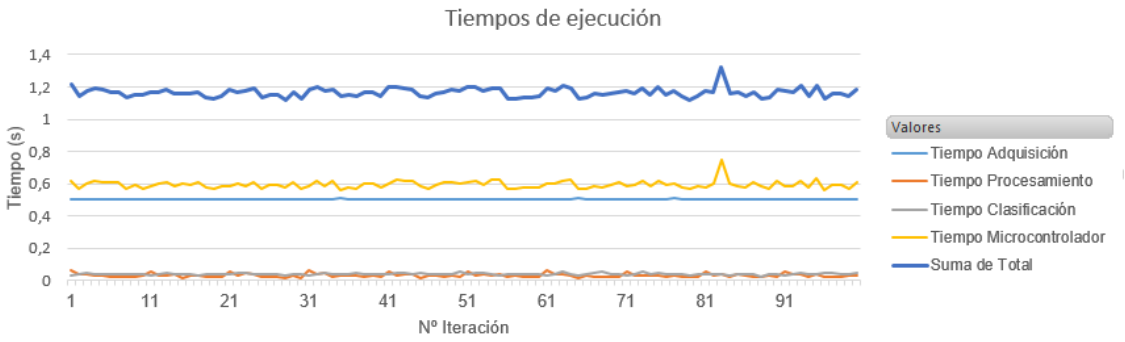


Fig. 7.5: Gráfica de tiempos de ejecución

En la gráfica de la Fig.7.5 se observa que la etapa de adquisición presenta un tiempo promedio de 0.5s, el cual fue el tiempo de muestreo indicado. También, se observa que la etapa que más tiempo demanda es la del microcontrolador. A modo de conclusión de estos tiempos se presenta la Tabla 7.1, donde se expone el tiempo promedio para cada una de las etapas así como el porcentaje que representa sobre el tiempo total de ejecución.

	Tiempo promedio	Porcentaje
Adquisición	0,502 s	43%
Procesamiento	0,029 s	3%
Clasificación	0,039 s	3%
Microcontrolador	0,595 s	51%
Total	1,165 s	100%

Tabla 7.1: Tiempos de ejecución

Tras el análisis de los tiempos de ejecución, se observa que el tiempo de respuesta ha sido optimizado notablemente durante el proyecto pero, a pesar de ello, no sería suficiente para realizar las funciones de una prótesis de mano.

7.3. Funcionamiento de la maqueta

En este apartado, se muestra el funcionamiento de la maqueta completa mediante imágenes. En la Fig. 7.6 se muestra la maqueta con la pinza abierta y, en la Fig.7.7 con la pinza cerrada. Por último, en la Fig. 7.8 se observa la pinza agarrando un paraguas.

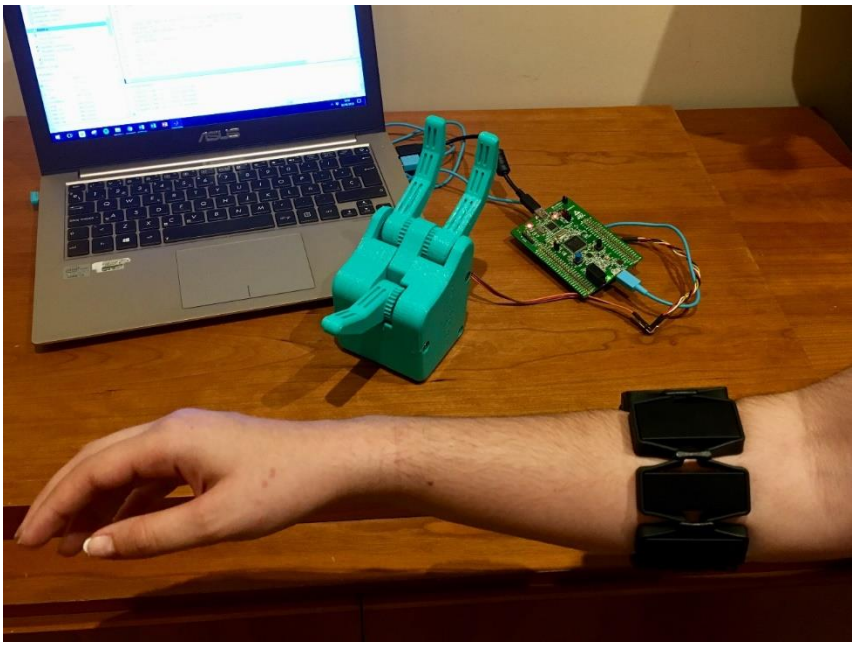


Fig. 7.6: Maqueta con pinza abierta

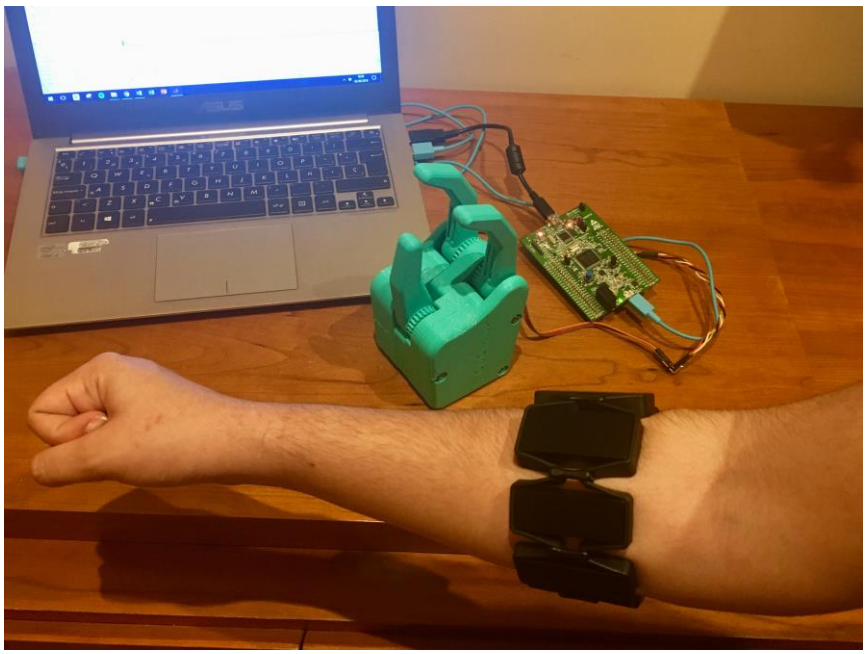


Fig. 7.7: Maqueta con pinza cerrada

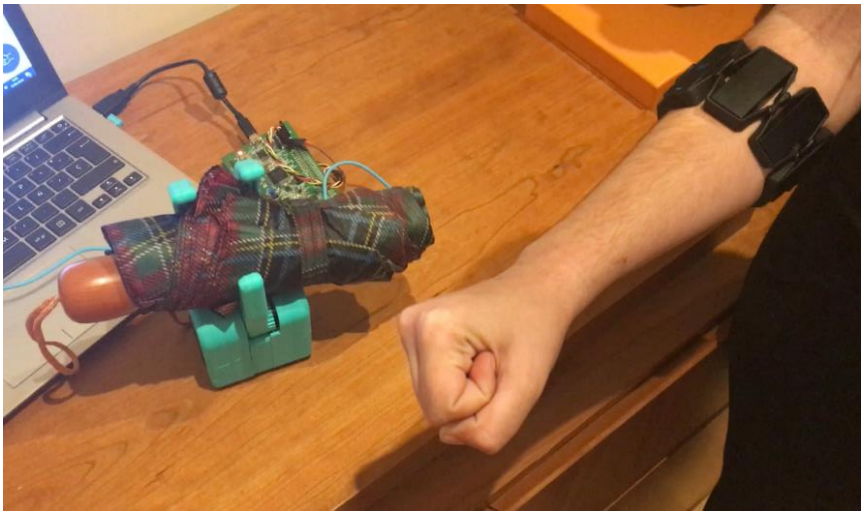


Fig. 7.8: Pinza agarrando un paraguas

Como se puede observar en las imágenes anteriores, la maqueta replica los movimientos de apertura y cierre correctamente. Además, se ha probado a agarrar un objeto, como muestra de la posible aplicación de la maqueta como una prótesis muy sencilla.

Por tanto, se puede concluir que la maqueta es completamente funcional y permite visualizar de manera muy clara la clasificación de movimientos realizada en el desarrollo del proyecto.

8. Conclusiones y trabajos futuros

En el capítulo introductorio de esta memoria se definieron una serie de objetivos, los cuales se han cumplido con éxito en el desarrollo del proyecto.

En primer lugar, se ha realizado la adquisición de señales EMG en MATLAB mediante la modificación del SDK del Myo Armband, así como el procesamiento de las mismas, segmentándolas, extrayendo características y reduciendo la dimensionalidad. De esta manera, se ha podido generar y entrenar una red neuronal capaz de clasificar los gestos de apertura y cierre de la mano realizados por el usuario.

También, se ha buscado el diseño para fabricar una pinza. El diseño elegido consta de tres dedos y es capaz de replicar los movimientos de apertura y cierre realizados por el usuario a tiempo real.

Como intermediario entre el ordenador y la pinza, se ha programado un microcontrolador, capaz de transformar el resultado obtenido tras la clasificación en una señal PWM que se transfiere al servomotor, que es el encargado de mover la pinza mediante un sistema de engranajes.

Una vez realizada la maqueta, se ha comprobado que es completamente funcional, por lo que se ha elaborado un guion de prácticas que cumple con el objetivo docente de aplicar la maqueta en sesiones de laboratorio.

Este proyecto estaba focalizado en la aplicación didáctica del mismo, por lo que al escoger el diseño de la pinza se decidió que fuera sencilla y fácil de fabricar, suficiente para ser capaz de replicar los movimientos analizados en este trabajo. Pero, al utilizar el reconocimiento de patrones se pretendía que el estudio realizado en este proyecto se pudiera aplicar en el control de una mano completa ya que, al haber utilizado esta técnica, bastaría con entrenar una red neuronal con más datos para poder identificar un mayor número de movimientos.

Con la aplicación en una mano completa, se podría utilizar tanto en teleoperación en manipulación robótica como en el control de prótesis robóticas en amputados que cuenten con algún miembro residual donde situar el Myo Armband.

Por tanto, se plantean como futuros trabajos: en primer lugar, optimizar los tiempos de ejecución, con el objetivo optimizar el sistema para, así, poder aplicarlo posteriormente en prótesis reales. En segundo lugar, realizar un diseño de mano completa y, por último, realizar la integración del sistema de control optimizado con el nuevo diseño.

También, sería interesante lograr la conexión del Myo Armband con Simulink, para poder realizar todo el proceso de manera gráfica.

Como conclusión final, se considera que el presente proyecto sienta las bases para el desarrollo de numerosos sistemas activados y controlados mediante señales EMG y reconocimiento de patrones gracias al uso del Myo Armband.

Bibliografía

[1] Universidad Carlos III de Madrid, "Aplicaciones de la automática en biomedicina", *Universidad Carlos III de Madrid*, 24/01/2018. [En línea]. Disponible en: https://aplicaciones.uc3m.es/cpa/generaFicha?est=223&asig=14061&idioma=1. [Acceso: 03/06/2018]

[2] A. Sánchez Anillo, "Matriz de electrodos EMG para detección de intención de movimiento de la mano", Trabajo fin de grado, Dpto. de Sistemas y Automática, Universidad Carlos III de Madrid, Madrid, España, 2017.

[3] *SENIAM*, Surface ElectroMyoGraphy for the Non-invasive Assessment of Muscles. Disponible en: http://www.seniam.org/. [Acceso: 03/06/2018]

[4] *Productos sanitarios de la Directiva 93/42/CEE*, Agencia Española de Medicamentos y Productos Sanitarios. [En línea]. Disponible en: https://www.aemps.gob.es/productosSanitarios/prodSanitarios/home.htm. [Acceso: 03/06/2018]

[5] Instituto Nacional del Cáncer, "Definición de electroencefalograma", *Instituto Nacional del Cáncer*. [En línea]. Disponible en: https://www.cancer.gov/espanol/publicaciones/diccionario/def/electroencefalograma. [Acceso: 03/06/2018]

[6] "Introducción a las pruebas de neuroimagen y neurofisiología más utilizadas en la clínica", Neurociencia Cognición. [En línea]. Disponible en: https://neurocienciacognicion.files.wordpress.com/2013/06/neurofisio-y-neuroimagen.pdf. [Acceso: 03/06/2018]

[7] J.Clausen et al., "Help, hope, and hype: Ethical dimensions of neuroprosthetics", *Science*, vol. 356, n.º 6345, pp. 1338-1339, jun. 2017. [En línea]. Disponible en: http://science.sciencemag.org/content/356/6345/1338.full. [Acceso: 03/06/2018]

[8] C. A. Quinayás Burgos, "Contribución al desarrollo y control de prótesis de mano", Tesis doctoral, ´Dpto. de Línea de investigación en Robótica Médica, Universidad del Cauca, Colombia, 2015. [En línea]. Disponible en: http://www.unicauca.edu.co/doctoradoce/publicaciones/Monografia_Quinayas.pdf. [Acceso: 03/06/2018]

[9] J. N. Eylis, "Prosthetic limbs controlled by RFID microchips", *Neuratheylis*, 30/03/2015. [En línea]. Disponible en: https://neuratheylis.wordpress.com/tag/rfid/. [Acceso: 03/06/2018]

[10] "How the i-limb works", *Touch Bionics*. [En línea]. Disponible en: https://www.touchbionics.com/products/how-i-limb-works. [Acceso: 03/06/2018]

[11] G. Ghazaei et al., "Deep learning-based artificial vision for grasp classification in myoelectric hands", J. Neural Eng, vol. 14, n.º 3, may. 2017. [En línea]. Disponible en: http://iopscience.iop.org/article/10.1088/1741-2552/aa6802/meta;jsessionid=440CAFE830BF3682EF472808E158E9FC.c3.iopscience.cld.iop.org. [Acceso: 03/06/2018]

[12] Newcastle University, "Hand that sees offers new hope to amputees", *Newcastle University,* 3/05/2017. [En línea]. Disponible en: https://www.ncl.ac.uk/press/articles/archive/2017/05/handthatsees/. [Acceso: 03/06/2018]

[13] R.Reiter, "Eine neue Elektrokunsthand", Grenzgebiete der Medizin, vol. 1, nº. 4, pp. 133-135, 1948.

[14] "Figure 3. Electric powered hand used by Reiter", Historical Aspects of Powered Limb Prostheses. [En línea]. Disponible en: http://www.oandplibrary.org/cpo/1985_01_002.asp?searchquery=reiter. [Acceso: 03/06/2018]

[15] "Russian prosthetic arm", *TECHNICAL RIGHT BELOW ELBOW AMPUTEE ISSUES.* [En línea]. Disponible en: http://www.swisswuff.ch/tech/?p=2366 [Acceso: 03706/2018]

[16] B. Hudgins, P. Parker, R. N. Scott, "A new strategy for multifunction myoelectric control", *IEEE Transactions on Biomedical Engineering*, vol. 40, n.º 1, pp. 82-94, ene. 1993. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/204774/. [Acceso: 03/06/2018]

[17] "Mano Bebionic", *Ottobock*. [En línea]. Disponible en: https://www.ottobock.es/protesica/miembro-superior/sistemas-de-brazo-y-mano/bebionic/. [Acceso: 03/06/2018]

[18] "i-Limb ultra", Touch Bionics. [En línea]. Disponible en: https://www.touchbionics.com/products/active-prostheses/i-limb-ultra. [Acceso: 03/06/2018]

[19] "Bionicohand", *Bionicohand.* [En línea]. Disponible en: https://bionico.org/. [Acceso: 03/06/2018]

[20] D.Copaci, "Informe 1 – Biomecánica del brazo humano", Prácticas de Aplicaciones de la Automática en Biomedicina, Dpto. de Sistemas y Automática, Universidad Carlos III de Madrid, 2018.

[21] D.Copaci, "Informe 2 – Procesamiento de señales de electromiografía", Prácticas de Aplicaciones de la Automática en Biomedicina, Dpto. de Sistemas y Automática, Universidad Carlos III de Madrid, 2018.

[22] "LSBIO_14-15.PDF", Laboratorio de Señales Biomédicas, Máster en Ingeniería Biomédica, Universidad Politécnica de Madrid, 2015. [En línea]. Disponible en: http://docencia.gbt.tfo.upm.es/wp-content/uploads/2014/12/LSBIO_14-15.PDF. [Acceso: 03/06/2018]

[23] "FACULTAD DE MEDICINA - LABORATORIODEFISIOLOGIA_AGOSTO2016.pdf", Dpto. de Fisiología, Facultad de Medicina UNAM, 2016. [En línea]. Disponible en: http://www.facmed.unam.mx/deptos/fisiologia/docs/LABORATORIODEFISIOLOGIA_AGOSTO2016.pdf. [Acceso: 03/06/2018]

[24] R. M. Enoka, "Muscle and Motor Units", en *Neuromechanics of Human Movement*, 4th ed. United States: Human Kinetics, 2001, 205-249.

[25] "9.3.4. La union neuromuscular", *UniNet*. [En línea]. Disponible en: https://www.uninet.edu/tratado/c090304.html. [Acceso: 03/06/2018]

[26] A. Villoslada, "Design nad implementation of a myoelectric control system for a printable robotic hand", Trabajo fin de máster, Dpto. de Sistemas y Automática, Universidad Carlos III de Madrid, Madrid, España, 2012.

[27] "Electromiografía (EMG)", *DALCAME.* [En línea]. Disponible en: http://www.dalcame.com/emg.html#.WxQIWCDtbIV. [Acceso: 03/06/2018]

[28] "Cybernetic Robotics", *ARAS – Hi Tech Robotic Solutions*. [En línea]. Disponible en: http://aras.kntu.ac.ir/research-themes/cybernetic-robotics/. [Acceso: 03/06/2018]

[29] "Tech Specs", *Myo Gesture Control Armband*. [En línea]. Disponible en: https://www.myo.com/techspecs. [Acceso: 03/06/2018]

[30] "Myo SDK 0.9.0: Myo SDK Manual", *Thalmic Labs - Makers of Myo gesture control armband.* [En línea]. Disponible en: https://developer.thalmic.com/docs/api_reference/platform/index.html. [Acceso: 03/06/2018]

[31] J. G. Abreu, J. M. Teixeire, L. S. Figueiredo, y V. Teichrieb, "Evaluation Sign Language Recognition Using the Myo Armband", presentada en XVIII Symposium on Virtual and Augmented Reality (SVR), Brasil, 21-24 Jun. 2016. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/7517255/ [Acceso. 03/06/2018]

[32] M. A. Oskoei y H. Hu, "Myoelectric control systems – A survey", *Biomedical Signal Processing and Control*, vol. 2, n.º 4, pp.275-294, 2007.

[33] E.J. Rechy-Ramirez y H. Hu, "Stages for developing control systems using EMG and EEG signals: a survey", School of Computer Science and Electronic Engineering, Reino Unido, Informe técnico CES-513, 2011.

[34] A. Villoslada, "Implementation of an EMG-based hand gesture classifier", Jupyter Notebook Viewer. [En línea]. Disponible en: https://bit.ly/2sElbw1. [Acceso: 03/06/2018]

[35] "Reducción de la dimensionalidad", *INAOE – Ciencias computacionales*. [En línea]. Disponible en: https://ccc.inaoep.mx/~emorales/Cursos/NvoAprend/Acetatos/pca217.pdf. [Acceso: 03/06/2018]

[36] "Proportional myoelectric control", *Wikipedia*. [En línea]. Disponible en: https://en.wikipedia.org/wiki/Proportional_myoelectric_control. [Acceso: 03/06/2018]

[37] T. Felzer y B. Freisleben, "HaWCoS: The Hands-free Wheelchair Control System", en: *Proceedings of the Fifth International ACM SIGCAPH Conference on Assistive Technologies*, Edinbugh, Scotland, pp. 127-134, ACM Press, 2002.

[38] M. A. Cazorla Quevedo, "Un enfoque bayesiano para la extracción de características y agrupamiento en visión artificial", Tesis doctoral, Dpto. de Ciencia de la Computación e Inteligencia Artificial, Universidad de Alicante, Alicante, España, 2000.

[39] H. A. Romo, J. C. Realpe, y P. E. Jojoa, "Análisis de Señales EMG Superficiales y su Aplicación en Control de Prótesis de Mano", Avances en Sistemas e Informática, vol. 3, n.º 1, pp.127-136, jun. 2007. [En línea]. Disponible en: https://bit.ly/2Hg0jjY. [Acceso: 03/06/2018]

[40] A. B. Ajiboye y R. F. Weir, "A heuristic fuzzy logic approach to EMG pattern recognition for multifunctional prosthesis control", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 13, n.º 3, pp. 280-291, Sep. 2005. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/1506815/. [Acceso: 03/06/2018]

[41] F. Ruiz Rico, "Selección y ponderación de características para la clasificación de textos y su aplicación en el diagnóstico médico", Tesis doctoral, Dpto. de Lenguajes y Sistemas Informáticos, Universidad de Alicante, Alicante, España, 2013. [En línea]. Disponible en: https://rua.ua.es/dspace/bitstream/10045/36215/1/tesis_fernando_ruiz_rico.pdf. [Acceso: 03/06/2018]

[42] M. A. Oskoei y H. Hu, "Support Vector Machine-Based Clasiffication Scheme for Myoelectric Control Applied to Upper Limb", *IEEE Transactions on Biomedical Engineering*, vol. 55, n.º 8, pp. 1956-1965, Ago. 2008. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/4463647/ [Acceso. 03/06/2018]

[43] D.J. Matich, "Redes Neuronales: Conceptos Básicos y Aplicaciones", Cátedra, Dpto. de Ingeniería Química, Universidad Tecnológica Nacional, Rosario, Argentina, 2001. [En línea]. Disponible en: https://bit.ly/2dN0ypN. [Acceso: 03/06/2018]

[44] E.N. Sánchez Camperos y A. Y. Alanís García, Redes neuronales: conceptos fundamentales y aplicaciones a control automático, 1ªed. Madrid: Pearson, 2006.

[45] C. Calderón-Cordova et al., "EMG Signal Patterns Recognition based on Feedforward Artificial Neural Network Applied to Robotic Prosthesis Myoelectric Control", presentada en Future Technologies Conference (FTC), United States, 6-7 Dic. 2016. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/7821705/. [Acceso: 03/06/2018]

Anexo A: Planificación y presupuesto

A.1. Planificación

La elaboración del proyecto se ha llevado a cabo en siete fases, las cuáles se muestran a continuación.

1. Planteamiento del problema: 8 horas

2. Estudio previo y búsqueda de información: 40 horas

3. Desarrollo MATLAB
   3.1. Adquisición de señales: 16 horas
   3.2. Procesamiento de señales: 18 horas
   3.3. Clasificación: 16 horas
   3.4. Programación del microcontrolador: 14 horas

4. Fabricación de la pinza
   4.1. Impresión 3D: 18 horas
   4.2. Montaje: 12 horas

5. Pruebas: 35 horas

6. Mejoras: 20 horas

7. Redacción de la memoria: 150 horas

Fases	Horas
Planteamiento del problema	8
Estudio previo y búsqueda de información	40
Desarrollo MATLAB	64
Fabricación de la pinza	30
Pruebas	35
Mejores	20
Redacción de la memoria	150
Total	347

Tabla A.1: Desglose de horas por fase

Semanas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Planteamiento del problema																						
Estudio previo y búsqueda de información																						
Desarrollo MATLAB																						
Adquisición de señales																						
Procesamiento de señales																						
Clasificación de movimientos																						
Programación microcontrolador																						
Fabricación de la pinza																						
Impresión 3D																						
Montaje																						
Pruebas																						
Mejoras																						
Redacción de la memoria																						

Tabla A.2: Diagrama de Gantt

A.2. Lista de material

En este apartado, se muestran los listados de materiales separados por Software y Hardware.

A.2.1. Software

Concepto	Cantidad
Windows 10 Home	1
Myo Connect	1
MATLAB R2017a - academic use	1
Neural Network Toolbox	1
Slic3r	1
Microsoft Office Hogar y Estudiantes 2013	1

Tabla A.3: Lista licencias Software

A.2.1. Hardware

Concepto	Cantidad
Ordenador	1
RS PRO 893-7310	8
Barra metálica 1mx4mm	1
Tornillo M4x20	2
Tornillo M4x30	2
Tornillo M3x12	5
Tuerca M3	11
Tuerca M4	4
Varilla roscada 3mmx200mm	1
Corona DS558MG Metal Gear Digital Servo	1
Plástico ABS	100g
Impresora 3D	1
Myo Armband	1
Adaptador Bluetooth	1
Cable Micro USB	1
Clips de fijación	10
Microcontrolador STM32F4DISCOVERY	1
Cable Mini USB	1

Tabla A.4: Lista materiales hardware

A.3. Presupuesto

A continuación, se detalla el coste del proyecto, separado en licencias de software y equipos, materiales y personal, así como un resumen de los costes totales.

A.3.1. Licencias de software y equipos

Concepto	Precio unitario	Duración licencia (meses)	Duración uso (meses)	Precio total
Windows 10 Home	145,00 €	12	6	72,50 €
MATLAB R2017a - academic use	69,00 €	60	6	6,90 €
Neural Network Toolbox	20,00 €	60	6	2,00 €
Microsoft Office Hogar y Estudiantes 2013	149,00 €	72	6	12,42 €
Ordenador	718,00 €	72	6	59,83 €
Impresora 3D	420,00 €	60	0,25	1,75 €
TOTAL				155,40 €

Tabla A.5: Presupuesto de licencias de software y equipos

A.3.2. Materiales

Concepto	Cantidad	Precio unitario	Precio total
RS PRO 893-7310	8	3,54 €	28,32 €
Varilla redonda aluminio 4mm 1m	1	1,55 €	1,55 €
Tornillo M4x20	2	0,07 €	0,13 €
Tornillo M4x30	2	0,10 €	0,20 €
Tornillo M3x12	5	0,06 €	0,29 €
Tuerca M3	11	0,03 €	0,31 €
Tuerca M4	4	0,05 €	0,20 €
Varilla roscada aluminio 3mmx200mm	1	2,20 €	2,20 €
Corona DS558MG Metal Gear Digital Servo	1	13,90 €	13,90 €
Plástico ABS	100g	0,04 €	3,60 €
Kit Myo Armband	1	178,43 €	178,43 €
Microcontrolador STM32F4DISCOVERY	1	16,83 €	16,83 €
Cable Mini USB	1	3,25 €	3,25 €
TOTAL			249,21 €

Tabla A.6: Presupuesto de materiales

A.3.3. Personal

Personal	Precio/hora	Horas	Precio
Tutor, Doctor en Ingeniería	50	40	2.000,00 €
Estudiante de Ingeniería	10	347	3.470,00 €
TOTAL			5.470,00 €

Tabla A.7: Presupuesto de personal

A.3.4. Resumen del presupuesto

Concepto	Precio
Licencias de software y equipos	155,40 €
Materiales	249,21 €
Personal	5.470,00 €
TOTAL	5.874,61 €

Tabla A.8: Resumen del presupuesto